Chorale Harmonization in the Style of J.S. Bach
A Machine Learning Approach

Alex Chilvers ALEX.CHILVERS@MQ.EDU.AU
Macquarie University, NSW 2109 North Ryde, Australia

Menno van Zaanen MVZAANEN@UVT.NL
Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands

Abstract

In this article, we present a new approach to the task of automatic chorale harmonization. Taking symbolic musical transcripts as input, we use the TiMBL Machine Learning package to generate harmonizations of unseen chorale melodies. Extracting features from the soprano notes only, the system generates the other three melody lines in the chorale. We experiment with features denoting different properties of notes, such as pitch, note length, location in the bar or piece, and features of surrounding notes, but also with global features such as metre and key. The current system harmonizes 41.71% of the data exactly like J.S. Bach did (modulo octave).

1. Introduction

Automatic harmonization, where accompanying melody lines are generated given a single melody line, is one of the oldest topics to be explored by computer musicians. A relatively early example of rule-based automatic harmonization using a constraint-based system is described in Pachet and Roy (2001). Other approaches such as the use of probabilistic finite state grammars (Conklin & Witten, 1995) or neural networks (Hild et al., 1992) have also been explored. Obviously, there is much more literature, but space restrictions do not allow us to provide more.

In this article we will use chorales composed by Johann Sebastian Bach. These chorales have long been the popular choice as data for computer music researchers exploring the harmonization challenge. The simple fact that there are many of these relatively short pieces, all composed in Bach's recognizable style, makes them appealing for this task.

Furthermore, it is known that Bach began with another composer's melody. This melody is used as the soprano voice in the chorales. He then harmonized this melody by adding notes for the alto, tenor, and bass voices, effectively adding his own accompaniment.

In the machine learning (ML) task described here, the same approach is used. Starting from properties extracted from the original melody, the soprano line, we apply a ML classifier, which outputs notes for the other three melody lines.

2. Approach

We treat the task of automatic harmonization as a classification task, which means that we can use standard ML techniques. As input, notes from the soprano melody are used. The output is a chord (i.e. notes for the three additional melody lines) describing the harmonization.

There are many other ways to approach the task. For example, one could classify each voice separately. However, initial experiments using that approach produced discouraging results.

2.1. Classifications

The output of the system is encoded using 12-bit strings, where a single bit represents whether or not the corresponding note is present in the chord. The output is represented relative to the chorale's tonic to avoid data sparseness problems. This means that the first bit in the bit string corresponds to the tonic, the second bit denotes a semi-tone higher than the tonic, and so on. For example, a major triad is represented by the string `100010010000`. Due to the three-voice nature of chorale harmonization, each bit string can contain a maximum of three `1`s.

Using this encoding, the information on the exact oc-

tave of a note is lost. Also, the ordering of voices is not explicitly encoded, as all notes are put in a bit vector which orders notes with respect to the tonic.

2.2. Feature vectors

The classifier takes properties from each of the initial (soprano) notes as input. These properties are encoded as features in feature vectors. The choice of features has a major impact on the performance of the system, so we consider a number of different approaches and combinations.

Local features encode information of the soprano notes. We use pitch and duration. Pitch is represented using the semi-tonal distance from the tonic. For example, when the note is the tonic, the value is `0`, whereas a whole tone above the tonic has value `2`. Duration is encoded using a `1` for a semibreve (whole note), a `4` for a crotchet (quarter note) and so on.

Contextual features represent information of the note in context, for example, the location of a note in a bar and the location of a bar in a piece. We also use the previous and next `n` soprano notes. This `n` is the size of the context and can be varied in different experiments. In the experiments described here, `n` is set to three.

With the goal of producing a nicely flowing harmonization, we have experimented with the previous `m` classifications, i.e. generated chords, as features. Previous attempts have been made using Markov models (Biyikoglu, 2003) and probabilistic inference (Allan & Williams, 2005), suggesting that using harmonization of previous notes can help to predict chords. Initial experiments showed that previous classifications have a negative impact. Many of them are wrong, which means that many incorrect values are incorporated into the feature vectors.

Finally, we incorporated global features, that provide information on the entire piece. In these experiments, we have used whether the piece is major or minor, and the piece's metre.

3. Results

For the experiments, we took 230 J.S. Bach chorales. The performance of the system was measured by calculating accuracy of the classified chords.[1] The results were computed using 10-fold cross validation.

As a baseline, the majority class, which is the major triad, is used. This results in an accuracy of 8.71 with a standard deviation of 0.86.

For the ML approach, we used the TiMBL package (Daelemans et al., 2002), which provides optimized k-NN classification algorithms. However, other classifiers can be used as well.

The best TiMBL system produced an accuracy of 41.71 with a standard deviation of 5.80. This result was acquired using all of the previously mentioned features, except for the previous classification features.

4. Conclusion

In this article we showed that automatic harmonization can be treated as a ML classification task. The results, which are generated using pitch, duration, key, positional, and contextual features, greatly outperform the majority class baseline. However, the current experiments simplify the task by removing octave and voice information.

There are many extensions that may improve the current system. Firstly, since the best results are generated using (almost) all features, it may very well be the case that additional features will further improve results. Secondly, the accuracy measure only takes into account harmonization exactly as Bach composed it. The way accuracy is measured now, chords counted as incorrect may actually still be musically valid.

References

Allan, M., & Williams, C. K. I. (2005). Harmonising chorales by probabilistic inference. *Advances in Neural Information Processing Systems 17*. MIT Press.

Biyikoglu, K. M. (2003). A markov model for chorale harmonization. *Proceedings of the 5th Triennial ESCOM Conference*.

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

Daelemans, W., Zavrel, J., van der Sloot, K., & van den Bosch, A. (2002). *TiMBL: Tilburg memory-based learner* (Technical Report ILK 02-10). Tilburg University, Tilburg, the Netherlands.

Hild, H., Feulner, J., & Menzel, W. (1992). HARMONET: A neural net for harmonizing chorales in the style of J.S. Bach. *Advances in Neural Information Processing 4 (NIPS 4)*.

Pachet, F., & Roy, P. (2001). Musical harmonization with constraints: A survey. *Constraints*, *6(1)*, 7–19.

[1] We only count exactly matching chords and do not use other musically correct chords, to reduce subjectivity.