ENHANCING CHORD CLASSIFICATION THROUGH NEIGHBOURHOOD HISTOGRAMS

Johannes Reinhard, Sebastian Stober, and Andreas Nürnberger

Faculty of Computer Science
Otto-von-Guericke-University Magdeburg, D-39106 Magdeburg, Germany
{joreinha,stober,nuernb}@cs.uni-magdeburg.de

ABSTRACT

The chord progression of a song is an important high-level feature which enables indexing as well as deeper analysis of musical recordings. Different approaches to chord recognition have been suggested in the past. Though their performance increased, still significant error rates seem to be unavoidable. One way to improve accuracy is to try to correct possible misclassifications. In this paper, we propose a post-processing method based on considerations of musical harmony, assuming that the pool of chords used in a song is limited and that strong oscillations of chords are uncommon. We show that exploiting (uncertain) knowledge about the chord-distribution in a chord's neighbourhood can significantly improve chord detection accuracy by evaluating our proposed post-processing method for three baseline classifiers on two early Beatles albums.

1. INTRODUCTION

The development of space-saving high quality audio formats, increase in storing capacity on computer hard disks and fast internet connections have boosted the spread of audio files containing music on computer platforms all over the world. To describe this content in a way that enables indexing and eventually searching beyond tags and keywords, suitable mid- and high-level features are required. Such a feature describing the harmonic content of a song is the progression of chords. Commonly, a chord is defined as a set of tones played at the same time. However, a looser definition also allows for tones not played simultaneously to form a chord, provided that they are to be interpreted as *somehow belonging together*. This makes chord determination a challenge even if music is available in symbolic notation. The problem is even harder, if the music is only available in raw audio format, as the single tones must be extracted from the musical signal. Hence, for chord recognition from audio signals, recognition errors have to be expected and handled.

We roughly divide the general approach to chord detection into three steps: Feature extraction, chord classification, and post-processing. In the first step, suitable features need to be extracted from the raw data. These features can then be used by a classifier to predict a chord, while using more robust features can significantly improve the accuracy of the classification. The third step tries to correct unavoidable misclassifications caused, e.g., by the presence of percussive sounds or harmonics. Here, the inclusion of musical theory or simple musical principles can help discovering possible errors and help avoiding them. This generalisation of the process of chord recognition outlined above is suggested by us to ease analysis and comparison of different approaches as a whole and distinct parts of it individually. A thorough study of current methods in the field (presented in Section 2) shows that it is sensible to do so.

The main scope of this paper is to propose and evaluate a post-processing step that can be applied independently of the features and classifier used. The only requirement for the classifier is to provide confidence or probability values for each chord. In the following, we first discuss related work in Section 2. Section 3.1 and 3.2 briefly describe the feature extraction step and classification step, respectively, to outline the context in which our post-processing method is applied. The post-processing step is explained in detail in Section 3.3. Subsequently, Section 4 presents the results of a preliminary evaluation. Finally, we point out directions for future research in Section 5 and draw conclusions in Section 6.

2. RELATED WORK

Sheh and Ellis showed in [2] that for the purpose of chord recognition of real-world musical recordings the chromagram feature, as introduced by Fujishima [1] outperformed other features. At the same time, their approach, which was based on the Hidden Markov Model (HMM), allowed for improvements: Using the flat start initialisation for the Expectation Maximization (EM) training, they avoided to provide hand-labelled aligned training data, the preparation of which is time-consuming. However flat start's inherent assumption is simplifying, so their HMM was insufficiently trained, in particular with regard to the quite big chord alphabet of 127 chords, they tried to model. Thus, their recognition rates evaluated on two Beatles songs were poor. 18 other Beatles songs were used for training.

Bello and Pickens improved this approach [3]. They only

Approach of	Classifier	Post-Processing
Fujishima [1]	Scalar product/Euclidean distance with hand-tuned chord templates	none
Sheh, Ellis [2]	Single Gaussian (Σ purely diagonal), flat start EM-trained	Viterbi on EM-trained transition matrix (TM)
Bello, Pickens [3]	Single Gaussian, manually specified	Viterbi on meaningful initialised + EM-trained TM
Lee, Slaney [4]	Single Gaussian (Σ purely diagonal), supervised trained with aligned training data	Viterbi on supervised-trained TM
Maddage et al. [5]	3 Gaussians, supervised-trained initialisation (STI) + EM-trained	Viterbi on EM-trained TM + key- and measure-based enhancements
Shenoy, Wang [6]	Pattern matching (?)	key- and measure-based enhancements
Burgoyne, Saul [7]	Dirichlet, STI + EM-trained	Viterbi on manually specified TM
Yoshioka et al. [8]	Mahalanobis distance	hypothesis search (also considering bass tones & common chord progression patterns)
Papadopoulos, Peeters [9]	Single Gaussian (supervised-trained or manually specified) or Correlation Computation	Viterbi on EM-trained or supervised-trained TM or manually specified TM (2 variants)

Table 1. Classification and post-processing steps of other chord recognition systems

trained a part of the HMM, while the other part was initialised based on music theoretic considerations and was not changed. Hence, they avoided to train the HMM by the use of any labelled training data (neither aligned, nor not aligned) at all. Moreover, they applied beat detection to adjust the size of a chroma window to the time between two beats to overcome problems caused by transient components in the sound. Additionally, they used a simple approach to chroma preprocessing, considering slightly different tuning of instruments. Their results (tested on a much smaller chord alphabet) were considerably better. They evaluated their system on two early Beatles albums.

Lee and Slaney [4] also followed a HMM-based approach and presented a method of automatic generation of aligned labelled training data. They collected musical pieces that were available in a symbolic format. It is possible to recognise chords on this format, and they used a software system to solve this task. Chord labels suggested by the system were mapped to audio file synthesised from the symbolic format. The resulting annotated audio file was accepted as ground-truth. That way, they were able to generate a large amount of training data and used this data to train the HMM.

A HMM, employed in a similar fashion as in Sheh and Ellis' approach, is used by Maddage et al. [5]. Besides the use of a more fine-grained resolution of the chromagram, the modelling of one chord with three states, the use of a more complex output distribution and the provision of generated training samples for each chord, Maddage et al. incorporate a post-processing step to correct possible misclassifications. It requires the detection of measures (based on beats) and the detection of the key, which is done for a 16 measure long window. Chords not in the detected key are disallowed and are replaced by other chords with high probability or with the previous chord.

This post-processing step is similar to the one of Shenoy and Wang [6]. However those do not use a HMM to determine the likely chords but a simpler classifier described more detailed though not yet unambiguously in [10]. In contrast to Maddage et al., their correction is not based on probabilities, as those are not provided by their classifier, but on rules considering the available chords in a measure. Another difference is, that their detected key is valid for the whole song.

This is a 'simplifying assumption' that is 'especially limiting', claimed by Burgoyne and Saul in [7]. Hence they avoid this assumption and model each possible chord present in each possible key as a single state in a HMM. They use Dirichlet distribution to model the emission probabilities and manually set the large transition matrix based on a model of harmonic relationship between chords and keys. Their perception of chords and keys being inseparable properties of a given harmony is also assisted by Yoshioka et al. [8]. However their chord detection approach is quite different:

Yoshioka et al. start at the beginning of a song and progress beat by beat, meanwhile creating hypotheses about the key and chord progression of a song until the front beat. Likely hypotheses are followed while highly unlikely hypotheses are pruned after a while. At the end of the song the most probable path is chosen as the chord progression. The score for a certain chord's prediction in a hypothesis is calculated based on three facts: A matching of the chromagram with a chord template, the dominant bass tone and the compliance with common chord progression patterns.

A closer look at the approaches described above reveals that in a way they use musical knowledge to enhance a possible misclassification. We define a post-processing step to be contained in a chord detection system as whenever the prediction of the classification function is not the final output of system. In that way, all the systems can be interpreted as to contain a post-processing step. Table 1 shows the different post-processing steps in the right column and the classifiers

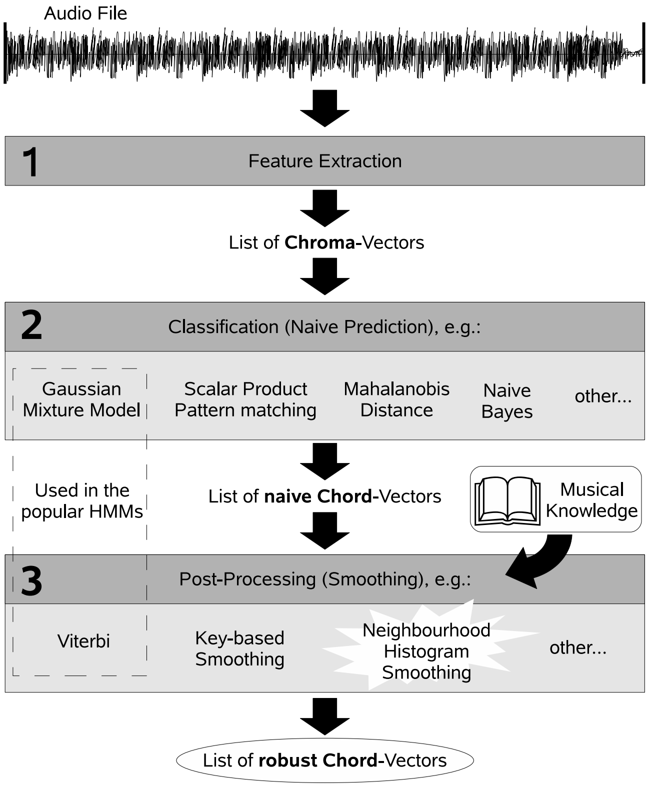
Audio File
1
Feature Extraction
List of Chroma-Vectors
2
Classification (Naive Prediction), e.g.:
Gaussian Mixture Model
Scalar Product Pattern matching
Mahalanobis Distance
Naive Bayes
other...
Used in the popular HMMs
List of naive Chord-Vectors
Musical Knowledge
3
Post-Processing (Smoothing), e.g.:
Viterbi
Key-based Smoothing
Neighbourhood Histogram Smoothing
other...
List of robust Chord-Vectors

Fig. 1. Generalisation of the chord recognition process

used in the middle column. Fig. 1 once again illustrates our generalisation of the chord recognition process. In the case of HMMs the post-processing is implicitly done in the Viterbi decoding. The Viterbi decoding is an algorithm that finds the optimal path in a sequence of states based firstly on the similarity of the observed chroma vector to the state output probability distribution and secondly on the transition probabilities contained in the transition matrix. See [11] for a tutorial to HMMs.

In a large-scale study [9], Papadopoulos and Peeters examine different choices of transition matrices for HMM-based chord detection. They show that manually set values based on music knowledge and set once for all songs to be evaluated are superior to those obtained by an unsupervised EM-training, which tries to get a sense of the harmonic movement of the song, that it is presently evaluated.

Although this holds for the application of the Viterbi, we still believe that the roughly extractable harmonic movement of a piece contains important knowledge which can and should be exploited to refine classification. Thus we propose an alternative post-processing step which integrates this knowledge. It is based on the histograms of chords in a chord's neighbourhood and does not need any information about the key of the song. It can be combined with any classifier as long as the classifier returns a score for every chord (i.e. not only the one most probable chord).

3. PROCESS OF CHORD RECOGNITION

Our approach to chord recognition is described in compliance with Fig. 1. The main contribution of our work is the histogram-based smoothing step (i.e step 3) presented detailed in Section 3.3. To show its universal applicability we tested it in on three different classifiers, which we describe in Section 3.2. First we explain the feature extraction in the following section.

3.1. Feature Extraction

The first step of the three-step approach is the extraction of the chromagram feature from an audio recording. For many approaches to chord recognition, this feature is still the feature of choice, though some report to get better results with features derived from it [12]. A chromagram is determined by transforming a signal into the frequency domain (either by a Short-Time Fourier Transform or a Constant-Q-Transform), and mapping the calculated intensities in the frequency bins to the pitches occurring in music. Subsequently, the pitches covering several octaves are mapped to just one octave, in a way that every pitch's intensity is added to the pitch class of its chroma (i.e. its musical tone), resulting in a 12 dimensional vector of pitch classes. This pitch class vector, also called the chromagram or just chroma, provides information about the intensities of the musical tones over the whole spectrum covered, missing the information of the octave heights of the tones. This is a sensible simplification, as it is assumed that a musical chord is composed of several tones, occurring in different octaves while the octave has no meaning for the detection of the chord.[1]

For the extraction of the chromagram features, we used the Sonic Visualiser [13] software. We decided to consider only a small frequency band of three octaves between 65,41 Hz and 523,25 Hz. Although this discards low bass tones and high voice or some high guitar tones, most important pitches for chord recognition are contained in this span. We also reduce the risk of wrong classification due to the presence of *misleading* harmonics, as only the lowest tones will have their misleading third or fifth or even higher harmonic in our pitch range. The second or fourth harmonic do not hamper chord recognition that much as they fall into the same pitch class as the fundamental.

We also account for instruments tuned deviant from the standard concert pitch of 440 Hz in a similar way as described in [14]. This step is quite important, especially for the early Beatles albums we evaluated on, as deviant tunings are common for these records.

Using a beat detection system included in the Sonic Visualiser software, the beat times of a song are extracted. We

[1] It should be remarked that this assumption is questionable as tones in the lower octaves, i.e. bass tones, may have higher influence on the chord's specification.

average the chromagrams located between two beats to form new chromagram vectors with larger and more meaningful window size. This common practice approach is based on the assumption that chords often change at beat times. As the time between two beats is longer than a chromagram window, averaging the windows between two beats will also do some smoothing.

Although the beat detection algorithm has problems in detecting the beats for some parts of songs correctly, we accept their recognition and do not correct them. We believe that errors in beat detection do not have bad influence on our results. Our simplifying assumption that chord changes occur at beats is altered to the assumption that chord changes occur at times with strong note onsets, which is one main feature the beat detection is based on.

3.2. Naive Prediction

In the second step, we use a classifier to predict the chord, solely based on the chromagram feature extracted in the previous step. We call this *naive prediction* to state the fact that this prediction might differ from our final prediction, which additionally incorporates knowledge of musical principles. For our approach, it is a requirement that the classifier does not only predict the most probable chord, but returns a probability or confidence for every possible chord considering the chromagram observed.

The chord alphabet we want to predict comprises just the 12 major and 12 minor chords. Other chords like diminished, augmented, seventh or other complex chords are mapped to major and minor depending on their third. E.g., an E diminished seventh chord would be mapped on an E minor chord as it contains a minor third.

We use three different classifiers in order to demonstrate that the post-processing technique described in the following section can be used in combination with arbitrary classifiers as long as they met the above stated requirement. The first classifier predicts a similarity score, calculating the scalar product between the chromagram vector and a chord template. The chord template contains a 1 if the tone is part of the chord and a 0 if it is not. So, if the order of tones in a chromagram vector is C,C♯,D,...,A♯,B, a C-Major chord template has the following format (1,0,0,0,1,0,0,1,0,0,0,0), see Fig. 2. For the C♯-Major chord the template looks similar with every number shifted to the right by one. Likewise a C-Minor chord template looks in the following way (1,0,0,1,0,0,0,1,0,0,0,0). Notice the minor third in contrast to the major third in the major chord template. This classifier, which is also used in [14], is simple but yields quite good results. Moreover, it does not only predict one chord but additionally gives a score for every chord. For our approach, each score is divided by the sum of all scores, so that it can be considered as a probability. It is denoted as $P(Chroma|c_i)\ \ i \in [1, 24]$ and describes the probability to observe the chroma vector $Chroma$ given chord c_i.

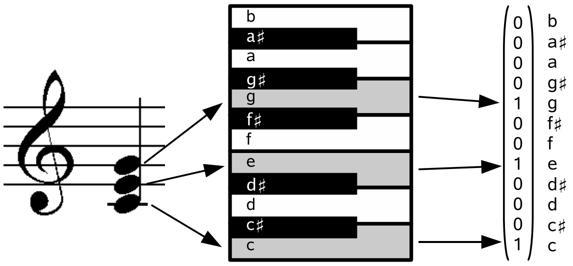

Fig. 2. The C-Major Chord Template

The second classifier calculates its values as the Mahalanobis distance between the chroma vector $Chroma$ and a distribution, represented by a mean vector μ and its associated covariance matrix Σ. The Mahalanobis distance is defined as:

$$d(Chroma, \mu) = \sqrt{(Chroma - \mu)^T \Sigma^{-1} (Chroma - \mu)}$$

The mean and covariance matrix are calculated from some training samples provided (see Section 4). This classifier is used in [8]. It is basically also used in those HMM approaches that use a single multivariate Gaussian to model their output distribution, which is done in [2], [3], [4] and [9], as the calculated distance and the calculated value of the output distribution can be transformed into each other (i.e. the ranking is the same). From the distance, we calculate a similarity measure as $sim = 1 - \frac{d}{d_{max}}$.

The third classifier is a naive bayes classifier[2]. As required for our approach, this classifier returns a probability for every chord. However, the probabilities from this classifier tend to be rather extreme. Often many chords have a probability close to 0, whereas only a small set of chords has high probabilities. This nature of the probability distribution makes this classifier less suitable. Nevertheless, we show that for sensibly chosen parameters, an improvement of recognition rates can still be achieved.

3.3. Smoothing

In this step we attempt to correct mistakes in the naive predictions obtained by the classifiers described in the preceding section. This smoothing step is motivated by the fact that music, to sound harmonically, does not have too many shifts, and repetition of musical patterns is more likely than steady change. This hypothesis is supported by the fact that most songs are written in a certain key, which constraints the choice of chords and can stabilise chord recognition. For a certain unknown chord, it is more likely to be one of the chords out of the pool of neighbouring chords than to be just an arbitrary other chord. Thus, we suggest to enhance chord recognition

[2] included in the Information Miner software developed at the Otto-von-Guericke-University in Magdeburg, `http://fuzzy.cs.uni-magdeburg.de/`

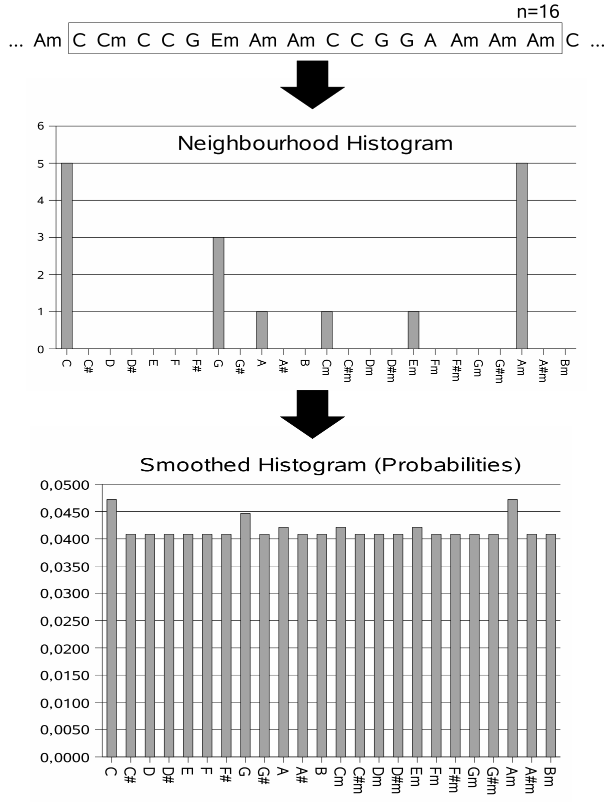

Fig. 3. Top: Naive predicted chord sequence (only most probable chord, i.e. $r = 1$); Middle: corresponding neighbourhood histogram; Bottom: Smoothed Histogram ($virtAppFactor = 2$, $relBonus = 0$)

by considering the chords in a certain neighbourhood. Therefore every chord is regarded as the center of a sliding window containing n chords. From these chords a histogram of the chord distribution is created including only the most probable chord for each chroma vector (see Fig. 3, Top and Middle). The bin-frequencies of the histogram are divided by n so that the histogram can be considered as the probability distribution of the chords in the neighbourhood, which we denote as $P(c_i)$. The new probability of each chord c_i given the respective chromagram $Chroma$ is calculated according to the bayesian theorem as:

$$P(c_i|Chroma) = \frac{P(Chroma|c_i) * P(c_i)}{P(Chroma)} \qquad (1)$$

Note, that no information about $P(Chroma)$ is required as it is only a normalisation constant to obtain values that sum up to 1. If only the rank but not the probability of a chord is interesting, it can simply be discarded, as it does not have any influence on the ranking of the chords. In case, a probability value is required, it is sufficient to divide all values obtained by discarding $P(Chroma)$ by their sum.

In this approach, the probability $P(c_i|Chroma)$ for chords that are not predicted at least once in the window is 0, because they have a marginal probability $P(c_i)$ of 0. Especially for small windows this appears to be too restrictive. To avoid zero marginal probabilities, one possibility is to add a constant number of *virtual appearance* to all bins which is calculated as $n * virtAppFactor$ (see Fig. 3, Bottom).

Alternatively, apart from the most probable chord, the next r most probable chords for each chroma vector could be added to the histogram as well. This, however, raises the question of how to weight the different chords. We define the value v_i to be added to the histogram for a chord ranked at position i as

$$v_i = \frac{P_{Rank=i} - P_{Rank=r+1}}{P_{Rank=1} - P_{Rank=r+1}}$$

where $P_{Rank=i}$ denotes the probability of the chord that is ranked at position i. If, for example, the 3 most probable chords have probabilities of 0.06, 0.052 and 0.05 and $r = 2$, then the bin frequency of the most probable chord is increased by 1, and the bin frequency of the second most probable chord is increased by 0.2.

Further, we define a measure of *reliability*, which is high if the prediction of the most probable chord is relatively certain and low if the difference to the second most probable chord is just marginal and thus the prediction is doubtful. It is defined as:

$$Rel = P_{Rank=1} * (P_{Rank=1} - P_{Rank=2})$$

All the reliability values of the predicted chords in a window are compared and a reliability bonus *relBon* added to the histogram bin of the most reliable chord. The bonus for the other chords decreases in the same manner as for ranked histogram probabilities, such that for the least reliable chord a bonus of 0 remains.

The smoothed chord change version of the song can then be smoothed further. While the factor $P(Chroma|c_i)$ in equation 1 is only based on our classification function and thus is not changed, the factor $P(c_i)$ is based on the histogram. This histogram, however, may have changed after the first smoothing step, as its new calculation is based on the values $P(c_i|Chroma)$. Thus further iterations of the smoothing step based on the updated histogram may further improve the results. The number of additional iterations is denoted with *k*.

All the above parameters influence the smoothing and we tested the impact of different parameter combinations on the performance of the chord prediction. Results are discussed in the next section.

4. EVALUATION

We tested our approach on the two early Beatles albums *Please Please Me* and *Beatles For Sale* - data which is also used for evaluation in [3], [12] and [14]. The ground truth chord annotations were kindly provided by C. Harte.

	Baseline accuracy	Average accuracy (all parameter combinations)	Accuracy (best parameters)
Please Please Me	56.42%	64.12%	69.05%
Beatles For Sale	63.91%	69.10%	74.02%
Overall	60.16%	66.61%	71.62%

Table 2. Accuracies before and after the smoothing step using scalar product classifier.

	Baseline accuracy	Average accuracy (all parameter combinations)	Accuracy (best parameters)
Please Please Me	54.60%	62.64%	72.52%
Beatles For Sale	62.81%	65.87%	75.56%
Overall	58.70%	64.25%	74.04%

Table 3. Accuracies before and after the smoothing step using the Mahalanobis distance based classifier.

Scalar product similarity classifier For the first classifier we investigated different parameter settings. Throughout our experiments, small window sizes yielded better results than larger ones. The best results were obtained with a window containing only 4 chromas. We tested the following parameters: window size $n \in \{4, 8, 16, 32\}$; factor multiplied by n to determine the number of virtual appearances in a histogram $virtAppFactor \in \{2, 5, 10\}$; worst rank of a chord to be included in the histogram $r \in \{1, 2, 3, 12, 24\}$; bonus value to be added to the bin of the most reliable chord $relBonus \in \{0, 1, 3\}$ and number of additional iterations $k \in \{0, 1, 2, 3, 4\}$. This meant an overall amount of 900 parameter combinations. The results are shown in Table 2. In general the post-processing led to an improvement of the detection accuracy. In only 4% of all tested parameter combinations the accuracy decreased. In average there was a relative increase by 10.72%. For the best parameter combination (4,5,3,1,4) there was a relative accuracy increase by 19.05%.

Mahalanobis distance classifier In contrast to the scalar product classifier, Mahalanobis distance classifier as well as the naive bayes classifier need to be trained on training data. We took our training samples from two other Beatles albums namely *With The Beatles* and *A Hard Day's Night*. We tested the second classifier with the same parameters as for the first. While the increase of accuracy for some promising parameters is again large, there is a higher risk of oversmoothing and thus decreasing the accuracy. This happened for 18 % of the cases. However for small window sizes of 4 or 8, the histogram smoothing performed well. The best accuracy with a relative increase of 26,13 % was reached with parameters (4,10,12,1,4) (see Table 3).

Naive bayes classifier As already stated in Section 3.2, we had to choose different parameters for the naive bayes classifier. This is due to the fact that for the naive bayes classifier, very few chords or just one chord dominates the probability distribution and has high distance to the following chords, so that it is unlikely that any correction will occur, if the classifier probabilities are multiplied with a highly smoothed histogram. Therefore, it is important not to donate too much initial appearance to the empty bins in the histogram, hence choosing a low $virtAppFactor$. This was approved by our tests where a $virtAppFactor$ of 0 performed best. The parameters tested for this classifier are: $n \in \{8, 16, 32, 128, 1000\}$, $virtAppFactor \in \{0, 0.5\}$, $r \in \{1, 3, 12\}$, $relBonus \in \{0, 2\}$ and $k \in \{0, 1, 2, 3, 4\}$. The best results were obtained independently of the choice of the parameters r and $relBonus$, with the other parameters set to $n = 1000$ (which meant that the 'neighbourhood' histograms comprise the chords of the whole song), $virtAppFactor = 0$ and $k = 2$ or $k = 3$. The results of our post-processing step for this classifier were not as good as for the other classifiers. This may be due to the worse baseline accuracy, so that the enhancements relied on less accurate data. The results are presented in Table 4.

Judging our results, we can see that our post-processing step significantly improves recognition accuracy which holds for three different classifiers we tested. However we also must compare our results to others. This is difficult as a credible comparison of post-processing methods requires the use of the same baseline classifier as well as the same chord alphabet to be considered and the same test set of songs. Maddage et al. and Shenoy and Wang at least provide accuracy numbers before and after their key-based post-processing. Shenoy and Wang also use the same chord alphabet as we do, but they do not evaluate on Beatles data. Although we avoid a direct comparison, we consider the accuracy rate of their system as a whole as very good. This may be partly due to their inclusion of measure detection. This is not yet included in our approach and might further improve our results. Apart from this, our approach does not make the limiting assumption that a song only contains chords belonging to a key but allows harmonic shifts. This may be an advantage as key changes (e.g. for the last repeated chorus) are not that uncommon in modern popular music.

Comparing our post-processing to the Viterbi decoding step of the HMM is also difficult. HMM-based publications do not provide the results of the classifier before the decoding,

	Baseline accuracy	Average accuracy (all parameter combinations)	Accuracy (best parameters)
Please Please Me	49.85%	52.49%	54.25%
Beatles For Sale	60.81%	63.98%	66.50%
Overall	55.02%	58.24%	60.37%

Table 4. Accuracies before and after the smoothing step using the naive bayes classifier.

as they consider it an inseparable part of the model. However Bello and Pickens' classifier (though not identical) is similar to our Mahalanobis classifier and the test set is the same. The results of both approaches only differ by 1% (their 75,04% to our 74,04%), so both methods seem to be nearly equally good. To get a reliable comparison we implemented the Viterbi by ourselves and get the same results: The Viterbi with a manually set transition matrix based on the double nested circle of fifth (see [9]) reaching 74,68% with the best chosen parameter ϵ was only marginal superior to our Histogram based smoothing. With a transition-matrix based on the correlation between key profiles [15] it performed with 70,54% considerably worse although it performed better than its double nested circle of fifth variant in the experiments in [9].

The good results of the Viterbi may be due to fact, that the transition matrix-based path search favours chord changes between chords of close harmonic relation. This is a sensible assumption and not yet integrated in our approach. However, Viterbi does not consider the roughly extractable harmonic movement of the piece, which in turn is integrated in our approach. We believe that combining both will further enhance recognition accuracies.

5. FUTURE WORK

As stated above, our approach does not account for the fact, that some chord changes are more probable than others. This knowledge can be derived from the theory of musical harmony and we currently study ways to integrate it into our approach.

To further improve our results, we also consider to include a measure detection and incorporate knowledge gained therefrom into our system. The good results of [5] and [10] showed the potential of this information. We also think of a special treatment of bass tone intensities, either in a classifier or in the smoothing step. The bass tone may be valuable information for the recognition of a chord.

As far as our work so far is concerned, we plan to further investigate relations and dependencies between the probability distribution output of the classifier, the choice of the parameters and their impact on the recognition rates. A way of normalising the classifier output distribution may make our smoothing step less sensitive to parameter settings.

To make analysis more substantiated, we plan to extend our evaluation. So far it only comprised two of 11 Beatles albums, which were chosen to compare our approach to others. Next, we intend to extend the evaluation on the full Beatles corpus. This may give insight in how our approach works on more sophisticated music, as the later Beatles albums got more complex in instrumentation as well as in musical harmonies.

6. CONCLUSIONS

In our paper, we present an approach for enhancement of chord detection accuracy. It works with a probability based classifier and uses the uncertain information of the chord distribution around a chord to aid a chord's prediction. It is assumed that for a chord to be determined, it is more likely to be from a pool of chords in the neighbourhood, than to be any other arbitrary chord. Though the idea is relatively simple, it shows promising results for three different classifiers we tested. However the absolute recognition rates of our approach are still improvable in our current state of work. We believe that improvements can be achieved by combining knowledge about the roughly extracted harmonic movement of the piece with the static knowledge about chord changes as stated by music theory.

We compared our approach to the Viterbi algorithm, which we consider to be the post-processing step of the widely-used Hidden Markov Models. Good results of both approaches show the potential and the need of a post-processing of a preliminary prediction of a classifier. This will make the chord recognition more robust and the chord progression of a song a more robust high-level feature to describe audio content for applications especially in the area of Music Information Retrieval.

7. ACKNOWLEDGEMENTS

The authors would like to thank Christopher Harte for sharing his high-quality annotation data for the complete Beatles corpus and the developers of the Sonic Visualiser tool. The bayesian classifier was kindly provided by Christian Borgelt and Matthias Steinbrecher.

8. REFERENCES

[1] Takuya Fujishima, "Realtime chord recognition of musical sound: A system using common lisp music," in

Proceedings of the International Computer Music Conference (ICMC), Beijing, 1999, pp. 464–467.

[2] Alexander Sheh and Daniel P. W. Ellis, "Chord segmentation and recognition using em-trained hidden markov models.," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, 2003.

[3] Juan Pablo Bello and Jeremy Pickens, "A robust mid-level representation for harmonic content in music signals.," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, 2005, pp. 304–311.

[4] Kyogu Lee and Malcolm Slaney, "Automatic chord recognition from audio using a supervised hmm trained with audio-from-symbolic data," in *AMCMM '06: Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, New York, NY, USA, 2006, pp. 11–20, ACM Press.

[5] Namunu Chinthaka Maddage, Changsheng Xu, Mohan S. Kankanhalli, and Xi Shao, "Content-based music structure analysis with applications to music semantics understanding," in *Proceedings of the 12th ACM International Conference on Multimedia*, 2004, pp. 112–119.

[6] Arun Shenoy and Ye Wang, "Key, chord, and rhythm tracking of popular music recordings," *Computer Music Journal*, vol. 29, no. 3, pp. 75–86, 2005.

[7] John Ashley Burgoyne and Lawrence K. Saul, "Learning harmonic relationships in digital audio with dirichlet-based hidden markov models," in *Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR)*, 2005.

[8] Takuya Yoshioka, Tetsuro Kitahara, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno, "Automatic chord transcription with concurrent recognition of chord symbols and boundaries.," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, 2004.

[9] Hélène Papadopoulos and Geoffroy Peeters, "Large-scale study of chord estimation algorithms based on chroma representation and hmm," in *Proceedings of the International Workshop on Content-Based Multimedia Indexing*, 2007.

[10] Arun Shenoy, Roshni Mohapatra, and Ye Wang, "Key determination of acoustic musical signals," in *Proceedings of the IEEE International Conference on Multimedia and Expo*, 2004.

[11] Lawrence R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[12] Kyogu Lee and Malcolm Slaney, "A unified system for chord transcription and key extraction using hidden markov models," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, 2007.

[13] Chris Cannam, Christian Landone, Mark B. Sandler, and Juan Pablo Bello, "The sonic visualiser: A visualisation platform for semantic descriptors from musical signals," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, 2006, pp. 324–327.

[14] Christopher A. Harte and Mark B. Sandler, "Automatic chord identification using a quantised chromagram," in *Proceedings of the 118th Audio Engineering Society's Convention*, 2005.

[15] Katy Noland and Mark Sandler, "Key estimation using a hidden markov model," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, 2006, pp. 121–126.