A Web-Based System For Suggesting New Practice Material To Music Learners Based On Chord Content

Johan Pauwels
Centre for Digital Music
Queen Mary University of London
j.pauwels@qmul.ac.uk

Mark B. Sandler
Centre for Digital Music
Queen Mary University of London
mark.sandler@qmul.ac.uk

ABSTRACT

In this demo paper, a system that suggests new practice material to music learners is presented. It is aimed at music practitioners of any skill set, playing any instrument, as long as they know how to play along with a chord sheet. Users need to select a number of chords in a web app, and are then presented with a list of music pieces containing those chords. Each of those pieces can then be played back while its chord transcription is displayed in sync to the music. This enables a variety of practice scenarios, ranging from following the chords in a piece to using the suggested music as a backing track to practice soloing over. We set out the various interface elements that make up this web application and the thoughts that went behind them. Furthermore, we touch upon the algorithms that are used in the app. Notably, the automatic generation of chord transcriptions – such that large amounts of music can be processed without human intervention – and the query resolution mechanism – finding appropriate music based on the user input and transcription quality – are discussed.

CCS CONCEPTS

• **Applied computing** → **Sound and music computing**; **Education**; • **Human-centered computing** → **Web-based interaction**.

KEYWORDS

music recommendation, music education, web application

ACM Reference Format:
Johan Pauwels and Mark B. Sandler. 2019. A Web-Based System For Suggesting New Practice Material To Music Learners Based On Chord Content. In *Joint Proceedings of the ACM IUI 2019 Workshops, Los Angeles, USA, March 20, 2019* , 4 pages.

1 INTRODUCTION

Regular practice is an intrinsic part of music education and absolutely necessary in order to improve. In a traditional educational context, music students are presented with new material in class and are then expected to rehearse on their own by repeating a small number of pieces. The reason for this limited number of pieces is purely practical. It is hard to find suitable material that is representative for a particular technique or musical concept without resorting to composing pieces especially for that purpose. Current music search interfaces are not very suitable for the process of finding practice material. While it is easy to find specific songs using meta-data or audio fingerprinting, discovering new music based on musical content is not directly supported.

However, repeating the same pieces over and over again can make practising monotonous and disengaging. At the same time, presenting the technique under study in more and varied contexts is likely to improve the effectiveness of the rehearsal. For these reasons, we are investigating how music pieces can be recommended to learners based on their choice of a set of chords as input. The main motivation behind the system we created is to answer the question: "What other music can I play with the chords I already know?"

Chords are chosen for querying because they are musically speaking on a high enough level of abstraction such that multiple pieces can match a query and because they make multiple practice scenarios possible. Users can for instance play along with the chords to practice transitioning between them or improve their improvisation skills. We believe this to be a novel type of music recommendation, as it is traditionally based on lower-level features or collaborative filtering [3]. Meanwhile, other applications of chord recognition are song-centric, meaning that you first decide which song you want to learn and then retrieve its chords, without chord-based recommendation [4].

In this paper, we present a web-based system that suggest music to users based on their selection of chords. We start by discussing the architecture of the application in section 2, followed by a breakdown of the various screens and user interface elements in section 3. We then move on to the algorithms involved in the back-end of the app in section 4 and end with a conclusion and some points for future work in section 5.

IUI Workshops'19, March 20, 2019, Los Angeles, USA

2 APPLICATION ARCHITECTURE

The web app consists of a client-side front-end written in plain JavaScript as a *single-page application*, such that transitions between the three views that the app is comprised of are seamless. The server-side back-end consists of two APIs that are called in sequence. The first is our own query API that returns suggested music pieces based on the user input. This API returns an ordered list of identifiers and their chord transcriptions. These identifiers are then passed on to the API of a music content provider to retrieve the corresponding metadata, artwork and streaming audio.

The front-end is built on the Bootstrap 3 framework[1] with a custom developed theme coded as SASS[2] rules. Our back-end is a light-weight API server built in Python using the Flask[3] framework. It is little more than a single – albeit complicated – call to a MongoDB[4] database containing the chord transcriptions, which then get returned as JSON[5].

3 USER INTERFACE ELEMENTS

As mentioned earlier, the user interface consists of three screens: a *query screen* where the user makes a selection of chords, a *list of results* presenting music pieces that contain the requested chords and a *music player* showing the chords in sync with the music for every piece. The current version of the interface was developed based on the user feedback and user observations we got from an earlier prototype [7]. Each of the three screens will now be presented in further detail.

Query Screen

As shown in Figure 1, users are presented with a grid of chord names on which they can click to add that chord to their selection (and click again to remove it). The grid contains sixty chord names, organised as the twelve possible roots in the columns and five chord types in the rows. The chord types are major, minor, dominant seventh, major seventh and minor seventh in that order such that simpler triads are displayed above more complex triads. These particular five chord types were chosen because they came out as the most popular types in an analysis of popular music [2] and together they cover almost the entire duration of that corpus.

Users can make a free selection of chord combinations. However, during our first trials of the interface, we noticed that users mostly wanted to select chords that go well together. Depending on their level of musical skill, recalling good combinations could involve some effort. Therefore,

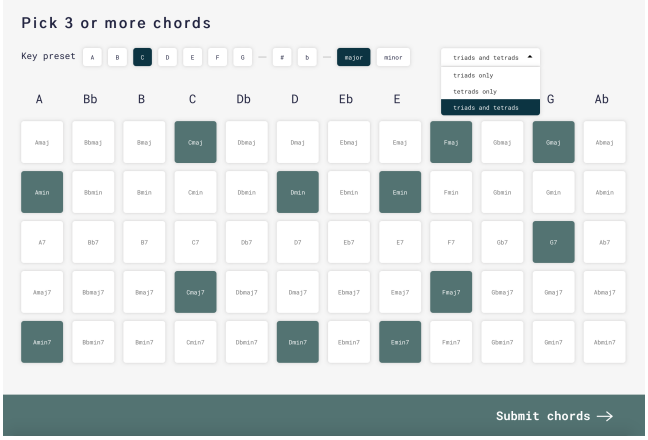

Figure 1: The query screen with the *Cmajor – triads & tetrads* preset activated.

and to minimise the number of clicks when selecting common combinations, we decided to offer presets that select all chords in a given key.

The available keys are formed by combining a major or minor mode with all combinations of seven natural tonics (A, B, C, D, E, F, G) and three accidentals (♮, ♭, ♯). For each of these keys, the diatonic triads, diatonic tetrads or a combination of both can be selected. In total this gives a total of $2 \times 7 \times 3 \times 3 = 126$ presets. After activating a key preset, the resulting selection can be edited further. For instance, you can start from the chords in a key and replace some that you don't want to practice with other ones not in that key. The usage of key presets is illustrated in Figure 1.

Selecting a key preset only results in a number of chords being activated. This key information is not passed in the query and therefore won't give different results compared to manually selecting the same chords, nor guarantee that the returned music is in that key. Notably, any major key preset gives the same results as its relative minor key.

Furthermore, for simplicity all chords are currently spelled with flats, both in the query screen and in the chord transcriptions that are returned as results. In the future, we plan to make the spelling adapt automatically to the selected key or to user preferences for sharps or flats when making a free selection. Similarly, user preferences for chord type formatting will be taken into account in the future, e.g. using △ instead of *maj7*.

Results List

After submitting a chord selection to the system, users are directed to the results screen shown in Figure 2. By clicking on individual results, users are taken to a player screen which allows to play back the song and its chords. Additionally, more results can be loaded (not depicted in the figure) as

[1] https://getbootstrap.com/
[2] https://sass-lang.com/
[3] http://flask.pocoo.org/
[4] https://www.mongodb.com/
[5] https://www.json.org/

Figure 2: The list of results for a particular chord query.

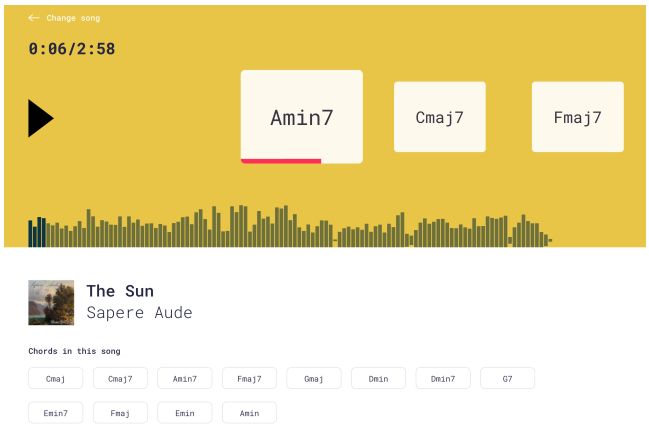

Figure 3: The music player displaying the chords in sync with the music.

long as there are more pieces corresponding to the query. Results are returned in batches of five by the query API, but the front-end checks whether the corresponding music pieces are still available from the content provider API before displaying them, which can result in smaller, irregular batch sizes.

The reason for the unavailability of a piece through the provider API is typically that the music is temporarily or permanently removed from the catalogue in the time between the indexation of the chord transcriptions and the user's query. Ideally, the indexation would keep track of changes in the catalogue and remove pieces from the chord transcription database when they disappear from the content provider, which would lead to a more consistent number of results being returned. However, for the time being the indexation requires manual intervention and has only been performed once at the start of development.

Finally, users can also return to the query screen in order to modify their chord selection.

Music Player

The player screen, as depicted in Figure 3, displays the chord transcription as text labels in boxes that scroll by in time with the music. The current chord label is displayed bigger and has a progress indicator at the bottom of its box. This allows users to anticipate the timing of a chord change and makes it easier to play along.

Below the chord boxes, a waveform is shown using the wavesurfer.js[6] library. It gives a wider overview of the current position with respect to the overall track duration and clicking on it allows you to move forwards and backwards through the music. In the future, we'd like to give a global overview of the chord changes by indicating them as coloured sections on the waveform.

Metadata and cover art of the piece is displayed under the music player followed by the set of chords appearing in the piece. The latter is shown because the chords of the piece do not necessarily correspond exactly to the chords selected for the query. The matching algorithm will try to find music that contains all of the requested chords, but failing that, might propose a partial match. Especially when a large number of chords is selected this might be the case.

Not shown in the image is that below the chords in the piece, the other results of the query are shown in a way similar to the initial list of results. This can be used to navigate easily to other results of the query. Alternatively, you can go back to the full list of results by clicking the text at the top of the page.

4 UNDERLYING ALGORITHMS

Two algorithms are working in tandem to suggest practice material to the users: an automatic chord estimator that generates chord transcriptions for a large dataset beforehand, and a query matching mechanism that finds appropriate pieces based on the transcriptions.

Chord Estimation Algorithm

The chord estimation algorithm is used to create a large collection of chord transcriptions. Because a large collection of transcriptions is needed to make all (or at least most) queries successful and creating this collection manually would be too time consuming, we resorted to automatic chord estimation. Furthermore, since the transcriptions need to be time-aligned to the audio, machine-readable and available for less popular music too, this makes it impossible to use

[6] https://wavesurfer-js.org/

any of the existing commercial or crowd-sourced chord sheet repositories.

However, the current state of the art for automatic chord estimation is far from perfect, so transcription errors can be expected. Therefore we use an algorithm [6] that outputs a confidence measure in addition to the chord transcription. This confidence in the quality of the transcription will be taken into account when suggesting music pieces to users.

The current version of the app is built by processing 100K tracks of the Jamendo[7] catalogue, a music platform that centralises Creative Commons music. They also provide an API[8] to retrieve metadata, cover art and streaming audio from. Work is ongoing to scale up the size of the underlying dataset and to include other music catalogues into the application.

Query Matching Algorithm

The query matching algorithm is running in realtime every time a query is made by a user. It aims to find music pieces that are the best match for the requested chords, but also balances this with the chord transcription quality as predicted by the confidence measure in order to give the best possible user experience. Details of the algorithm can be found in [5]. The decision was made to only return pieces that contain no other chords than the ones that present in the user selection. The reasoning is that beginners might not know how to play all possible chords yet and that returning pieces containing chords they do not know how to play would lead to a bad user experience. An option for more advanced musicians to include one or more wildcard chords in the query is planned for a future version.

A consequence of this decision is that the more chords are selected, the larger the subset of music pieces is that the system can suggest. Therefore the query screen includes the instruction to select at least three chords. This is not a technical requirement, but selecting a single chord would give the user music pieces consisting only of a single chord, not very suitable for practising, and therefore lead to an unsatisfactory experience.

Preference is given to music pieces that contain as much of the requested chords as possible. There might not be a music piece available in the dataset with the exact chord combination that's requested, especially if the selected chords do not go well together musically, but pieces with the largest subset of possible chords will be suggested in that case, all while considering the confidence to return high quality transcriptions.

We'd like to give users more control over the matching algorithm in the future, by letting them specify preferences for genre or difficulty (measured as the rate of harmonic change, for instance).

[7] https://www.jamendo.com/

[8] https://developer.jamendo.com/v3.0/docs

5 CONCLUSION AND FUTURE WORK

In this paper, we demonstrated a web-based application that recommends new music to learners for practising their instruments with those pieces. Users need to make a selection of chords and the system then returns a list of music pieces that contain them. A variety of practice scenarios are envisioned, from beginners playing along with the chords to more advanced learners playing arpeggios or soloing over the changes.

With this interface, we plan to perform an extensive user study to evaluate the quality of the underlying algorithms, both the automatic chord estimation and the query resolution, and the holistic system, including the user interface as a whole.

Further planned improvements include changing the format of the chord labels according to user preferences or depending on key, thereby adding support for transposing instruments. Another potential topic for research is how to assist users in the free exploration of chord selections and if providing visual hints for chord combinations that are common in the data would be useful for that.

In its current state, the system just recommends music to play along with, but it could be extended to a more complete music education app by letting it listen to the user's performance and return automatic scoring and feedback. The guitar-specific system presented in [1] may provide a way towards this goal.

REFERENCES

[1] Shunya Ariga, Masataka Goto, and Koji Yatani. 2017. Strummer: An interactive guitar chord practice system. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME).*

[2] John Ashley Burgoyne, Jonathan Wild, and Ichiro Fujinaga. 2011. An Expert Ground-Truth Set for Audio Chord Recognition and Music Analysis. In *Proceedings of the 12th ISMIR Conference.*

[3] Òscar Celma. 2010. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space.* Springer-Verlag Berlin Heidelberg.

[4] W. Bas de Haas, José Pedro Magalhães, Dion ten Heggeler, Gijs Bekenkamp, and Tijmen Ruizendaal. 2014. Chordify: Chord Transcription for the Masses. In *Proceedings of the 13th ISMIR Conference, Late Breaking and Demo Session.*

[5] Johan Pauwels, György Fazekas, and Mark B. Sandler. 2018. Recommending songs to music learners based on chord content. In *Proceedings of the 2018 Joint Workshop on Machine Learning for Music.*

[6] Johan Pauwels, Ken O'Hanlon, György Fazekas, and Mark B. Sandler. 2017. Confidence Measures and Their Applications in Music Labelling Systems Based on Hidden Markov Models. In *Proceedings of the 18th ISMIR Conference.* 279–285.

[7] Johan Pauwels, Anna Xambó, Gerard Roma, Mathieu Barthet, and György Fazekas. 2018. Exploring Real-time Visualisations to Support Chord Learning with a Large Music Collection. In *Proceedings of the 4th Web Audio Conference.*