UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

MARCELO CARDOSO BORTOLOZZO

Improving Rare Chord Recognition Through Self-Learning Techniques and Weak Label Generation

Thesis presented in partial fulfillment of the requirements for the degree of Master of Computer Science

Advisor: Prof. Dr. Claudio Rosito Jung
Coadvisor: Prof. Dr. Rodrigo Schramm

Porto Alegre
November 2022

CIP — CATALOGING-IN-PUBLICATION

Bortolozzo, Marcelo Cardoso

Improving Rare Chord Recognition Through Self-Learning Techniques and Weak Label Generation / Marcelo Cardoso Bortolozzo. – Porto Alegre: PPGC da UFRGS, 2022.

64 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2022. Advisor: Claudio Rosito Jung; Coadvisor: Rodrigo Schramm.

1. Automatic chord recognition, data imbalance, self-learning, focal loss, weak labels, sequential data. I. Jung, Claudio Rosito. II. Schramm, Rodrigo. III. Título.

ABSTRACT

In the context of Automatic Chord Recognition (ACR), the main goal is to extract and classify musical chords from sequential information (audio excerpts). It is a challenging task, not only when developing a classifier for it, but also when labelling its data. It requires a certain expertise in the domain when defining the ground truth for the data, unlike other tasks, such as object recognition. This results in a limited number of publicly available datasets and even more limited number of rare chord samples in general, resulting in a very biased performance by classifiers. In this work, some techniques to mitigate this issue will be explored. First, a modified loss function, known as focal loss will be applied, attempting to improve the performance of these rarer classes. Next, an image recognition training technique known as Noisy Student, which applies an iterative self-learning process to improve performance, will be modified for the audio domain and applied to the problem of ACR. Furthermore, an extension for this last technique using a weakly labeled generated ACR dataset for confidence boosting in the self-learning process will be proposed and applied. The dataset generation algorithm, based on data extracted from online musical chord communities will also be presented. The experiments performed showed significant improvements on the prediction accuracy of rare chords, while also slightly improving the overall accuracy for all chords in general.
Keywords: Automatic chord recognition, data imbalance, self-learning, focal loss, weak labels, sequential data.

Melhorando o Reconhecimento de Acordes Raros Através de Técnicas de Auto-Aprendizagem e Geração de Rótulos Fracos

RESUMO

No contexto de Reconhecimento Automático de Acordes (ACR), o principal objetivo é o de extrair e classificar acordes musicais a partir de um dado sequencial (trechos de áudio). Essa é uma tarefa desafiadora, não somente quando desenvolvendo um classificador para ela, mas também quando rotulando seus dados, já que é necessária uma expertise em seu domínio para poder definí-lo, diferentemente de outras áreas, como reconhecimento de imagem. Isso resulta em um número limitado de conjuntos de dados disponíveis publicamente, e um número ainda mais limitado de amostras de acordes raros, gerando resultados enviesados em classificadores. Neste trabalho, algumas técnicas para mitigar esse problema serão exploradas. Primeiro, uma função de loss modificada, conhecida como Focal Loss, será aplicada, buscando uma melhoria nessas classes mais raras. Em seguida, uma técnica de auto-aprendizagem do domínio de reconhecimento de imagem, conhecida como Noisy Student, será extendida ao domínio de áudio e aplicada ao problema de ACR. Além disso, uma extensão para essa mesma, utilizando um conjunto de dados com rótulos fracos e gerados automáticamente para aumentar a confiança do algoritmo de auto-aprendizagem, será proposta e aplicada. O algorítmo utilizado para a geração desse conjunto de dados, com base em dados extraídos de comunidades online para acordes musicais, também será apresentado. Os experimentos realizados trouxeram ganhos significativos de performance para os acordes raros, também gerando um pequeno ganho na performance de acordes em geral.

Palavras-chave: reconhecimento automático de acordes, dados desbalanceados, auto-aprendizagem, focal loss, rótulos fracos, dados sequenciais.

LIST OF ABBREVIATIONS AND ACRONYMS

ACR	Automatic Chord Recognition
ML	Machine Learning
MIR	Music Information Retrieval
maj	Major
min	Minor
CQT	Constant-Q Transform
HMM	Hidden Markov Models
DNN	Deep Neural Network
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
CRF	Conditional Random Field
DTW	Dynamic Time-Warping
PL	Predicted Labels
EL	External Labels
CFS	Chroma Feature Similarity
TIV	Tonal Interval Vector
WCSR	Weighten Chord Symbol Recall
CSR	Chord Symbol Recall
ACQA	Average Chord Quality Accuracy

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

Identifying and labeling musical chords is a complex task, not only for algorithms and classifiers but also for humans. Its scope comprises multiple different nuances and ambiguities that sometimes may even cause disagreements between expert musicians. Therefore, such a difficult task might not be so easily handled by Machine Learning (ML) algorithms.

In Computer Science, there is a field of study that specializes in extracting and labeling data related to music, known as Music Information Retrieval (MIR). It is in a sub-task of this field, known as Automatic Chord Recognition (ACR), that this thesis will provide a contribution. ACR usually consists of identifying a sequence of chords out of a musical audio excerpt. This means retrieving the onset and offset timestamps as well as the label that identifies the chord being played during that period of time. Although it seems like a straightforward classification problem, there are many possible chord labels, some of which are, as will be referenced from now on in this document, considered rare or complex chords.

These rare and complex chords can be described as such because they are underrepresented in most ACR datasets or even in music recordings, depending on their musical style. For instance, consider that most ACR datasets are mainly based on western popular music, which includes Rock, Pop, Jazz and Blues musical styles. This results in a set of chords being more common in some of these styles than others, and some of them not being common at all. Besides, these chords are sometimes defined in such a way that their structure is significantly more complex than their common counterparts. More details on the musical concepts for chords and ACR will be provided in Chapter 2. Also, it is important to mention that although these different styles are played using different sets of instruments, it is expected that ACR techniques such as this one provide an instrument-agnostic solution.

The capability of recognizing such chords has different kinds of applications both in educational and commercial contexts. For instance, consider a music learning application that shows the user a series of chords to be played and can recognize if it was played correctly, giving immediate feedback. Another example could be the transcription of any given music to a sequence of chords for it. In this case, any user would be able to extract the chords for a song they are listening to, making it easy for them to use this information. In this context, the so-called rare and complex

chords play a significant role, as they are often the ones harder to learn or to identify from audio, therefore providing good performance for them is necessary.

Current state-of-the-art algorithms for ACR have evolved from a knowledge-driven strategy to a data-driven one (PAUWELS et al., 2019). This means that in its beginnings, the classification process relied much more on pre-determined musical knowledge that would be imbued in an algorithm. This is no longer the case with data-driven systems: these latter solutions rely much more on the volume of labeled data made available for them to learn from, and not on the specific musical knowledge that its creator could provide to it.

However, in the current scenario of the ACR community, most publicly available datasets comprise no more than a few hundred recordings, creating a clear limitation for data-driven models based on ML. Besides, there is a strong imbalance issue in these datasets, as rare chords are a relatively small portion of the overall dataset composition. For example, Table 1.1 provides an overview of the chord class distribution in four datasets commonly used in ACR: Isophonics (HARTE, 2010), RWC (GOTO et al., 2002), Billboard (BURGOYNE; WILD; FUJINAGA, 2011), and USPOP (BERENZWEIG et al., 2004). As is shown in the table, there are two classes that stand out as always having the largest representation in the dataset: the maj and min chords, which are the common chords mentioned previously. Although some other chords may sometimes have a similar representation reaching more than 10% of the dataset, they mostly are a very small portion, in some cases representing less than 1%. This not only creates difficulty in scaling the learning capabilities of a classifier due to lack of data, but also creates a learning environment that is likely to be biased towards the more common chords if no care is taken when selecting the appropriate evaluation metrics.

	Isophonics	RWC	Billboard	USPOP
maj	50.4	45.1	50.4	56.1
min	12.6	14.7	12.6	14.6
7	10	7.2	10	6.4
min7	7.8	13.8	7.8	8.9
maj7	2.8	7.4	2.8	2.7
sus4	2.7	2.6	2.7	1.6
maj6	1.1	2.3	1.2	0.8
min6	0.3	0.3	0.2	0.8
9	1	0.3	0.4	0.8
dim	0.2	0.8	0.2	0.1
aug	0.1	0.4	0.1	0.1
hdim7	0.2	0.4	0.2	0.1
other chords[1]	10.8	4.7	11.4	7

Table 1.1 – Chord class distribution (in percentage) on different ACR datasets

If there is a clear need for more and more data, with new samples from each of the different classes, the first solution might be to work towards building new datasets. However, this solution is not one that scales well in solving the problem. In fact, creating a new ACR dataset is not a trivial task: it requires musical training, and such a task may not be done by any person, unlike, for instance, labeling common objects, which require no specialized knowledge, in an image.

It is in this context of the need for more easily accessible data and bias towards common chords that the work presented here is inserted. It will explore the use of different training techniques for reducing this bias, and also will attempt to increase the amount of available data for training through the usage of weak labeling techniques.

Weak Labelling is a technique in which noisy labels are used, and it has already been explored in this context of data-driven solutions (HAN et al., 2018). This work will also attempt to provide solutions for the lack of data available for ACR tasks by exploring a weak labeling technique based on the usage of chord information extracted from online guitar chord communities. These weak labels will, in turn, be used in conjunction with the techniques mentioned above to attempt to maximize the gain obtained from them.

Regarding the training process, the techniques that will be explored aim at providing a solution that could be applied to any classifier and mitigate the intrinsic bias existing in the datasets presented above. This will be done through the usage

[1] dim7, minmaj7, sus2, maj9, min9, 11, 13

of an adapted loss function, known as Focal Loss (LIN et al., 2017) and through the adaptation to the musical domain of a self-learning technique used in imbalanced image classification scenarios, known as NoisyStudent (XIE et al., 2020).

This work is organized as follows: Chapter 2 will provide a brief overview of the musical and ACR concepts relevant to its understanding; Chapter 3 will present a review of the existing work related to the one being presented here; Chapter 4 will outline the weak label dataset generation technique presented above; Chapter 5 will describe the different techniques and algorithms evaluated with the objective of improving the performance of rare chord recognition in ACR classifiers; Chapter 6 will provide the results and discussion of the experiments performed; and finally Chapter 7 will wrap up this work, by presenting its conclusion.

2 MUSIC THEORY AND AUTOMATIC CHORD RECOGNITION

This chapter provides a brief overview of two relevant concepts to be understood before diving deeper into the contributions of this work. The first one is related to basic music theory, particularly the definitions of chords, their relation to one another, and what common, rare and complex chords are. The second subject links these music theory concepts with ACR, and provide some important differences and parallels between other subjects where Machine Learning is applied in a similar way.

This chapter is essential, as the reader might not know the concepts presented here, and their understanding is indispensable for an easier understanding of what will be discussed in further chapters. Music theory, in itself, is outside the scope of this work, but much of the proposed contributions and pre-processing steps used in the experiments described were developed based on these concepts.

Besides, a brief introduction to ACR will also be important to identify similarities with other, more widespread techniques, such as image recognition or generic audio processing. This will allow for techniques from these related fields to be drawn upon in the experiments that follow. Also, it will allow for the differences between these fields to be described and addressed.

2.1 Musical Theory

This section presents some basic concepts and naming conventions extracted from music theory.

Pitch is a tone generated through vibration at a certain frequency: faster vibration means a higher pitch. Every time this frequency doubles, we say that there is an octave between these two pitches. Besides, these pitches have very similar perceived sounds, and therefore they are determined as being part of the same pitch class (LAITZ, 2022). For instance, let us consider Figure 2.1, which contains a drawing of a piano keyboard and its respective pitches. In particular, we highlight pitch class C and all the different pitches that are part of it. In musical notation, such pitches are represented by notes placed on certain positions of a score.

[1]Extracted from: <https://www.audiolabs-erlangen.de/resources/MIR/FMP/C1/C1S1_MusicalNotesPitches.html>

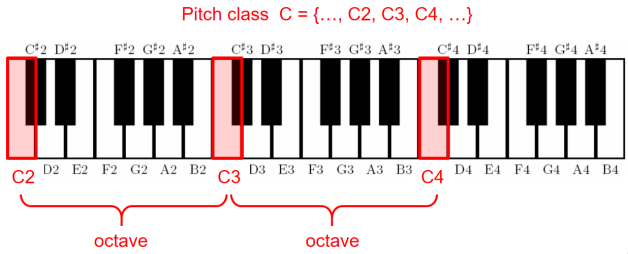


Figure 2.1 – Pitches and Pitch Classes on a piano.[1]

Whenever two pitches are played, they are called a dyad. Three pitches form a triad, four a tetrad, and so on. This paragraph handles a brief overview of chords, which is the main concept in musical theory that is the main focus in ACR problems. A chord is a set of pitches being played at the same time, or sometimes, sequentially in a short time span (e.g., arpeggios). Its lowest pitch, with the smallest frequency, is known as the root, and the remaining pitches can be identified based on this root in conjunction with a quality (LAITZ, 2022).

Chords are built based on the interval relations between the pitches that compose them, which are mostly based on an underlying musical scale, defining the chord type, or qualifier. This introduces two new concepts: the intervals and the scales. The interval is a name given to the relation between two pitches, based on their distance and frequency ratios. One example of an already seen interval is an octave, which has a frequency ratio of 2:1 and represents pitches from the same class. In between the two pitches of an octave, other kinds of intervals may be built, such as thirds, fifths, sixths, and so on, with some of them being major, minor, augmented, or diminished.

One important concept is that all of these intervals can be defined based on the number of semitones that compose them. A semitone is the smallest musical interval that is referenced in western tonal music, and in the scheme provided above, it represents the interval between two adjacent pitches (or keys in the piano keyboard). So, for example, an octave has 12 semitones, or 12 pitches between each end of the interval, a major third has four semitones, and a perfect fifth has 7. This allows for a clear definition of intervals and an easier way to understand how chord names are constructed.

The second important point is the notion of scales, which are a set of musical notes grouped together. For the context being presented here, there is no need to go into much detail, as it focuses more on western tonal music, and the chords existing in it, so a simple overview and its relation to chords will be presented. Scales

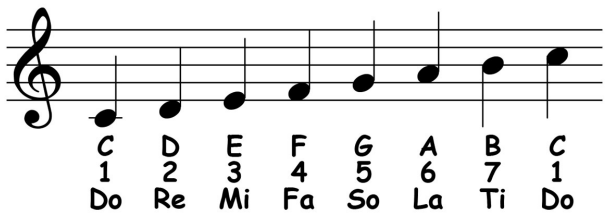


Figure 2.2 – C major scale

follow certain guidelines when being built, based on the interval relations between its components. For example, a major scale follows a certain sequence of semitone and tone (i.e., an interval of two semitones) between each of its pitch members, resulting in a set of 7 distinct notes. A natural minor scale follows another recipe, but is also very clearly defined. This allows for a scale to be built based on different starting points, resulting in the same scale type, as long as they follow the same rules. This has a tight relationship to chords, as most chord types are built based on the scale of its root pitch, as we'll see next.

Let us consider the `C:maj` chord, which has three pitches: the root itself (C), a major third interval above the root (i.e., the E note) and the interval of a fifth above the root (i.e., the G note). Any major chord will follow this exact same relationship between its pitches, and the reason for that is tightly coupled with the scale on which it is based. Let us take a look at figure 2.2, which presents the C major scale on a music score, along the note names and the sequential set of numbers, which represent the pitches degree of that scale. As we can see, the C, E and G notes are, respectively, the first, third, and fifth members of the scale. Considering that there is a predefined relationship between the intervals of any major scale, as long as we take the pitches from the same relative positions of other major scales, we'll always have a major chord. The same applies, for instance, to minor chords, which take the first, third and fifth pitches of the natural minor scale, which provides a different set of intervals between its pitches, therefore resulting in a different chord.

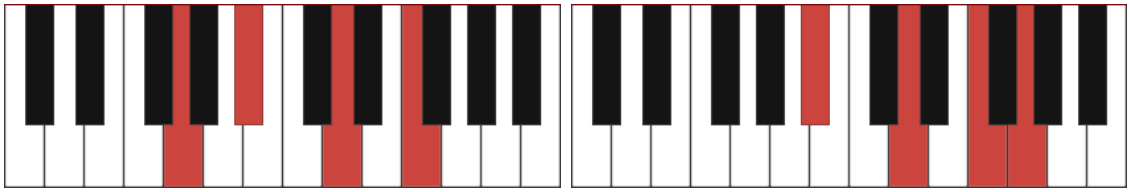
After providing a short overview of the required musical context, we provide some definitions of common, rare, and complex chords. Common chords, as will be defined from now on, are the major (maj) and minor (min) triads, which are by far the most common and also some of the simplest chords available, as they are

composed of only three pitches. Complex chords will be defined in this work as the ones with four or more pitches, while rare chords will simply be the ones that are usually underrepresented in datasets. We note that some chords are complex but might not be so rare, such as the seventh chord, while others are not complex but are usually quite rare, such as the augmented and diminished chords. Despite having different natures, rare and complex chords will be considered part of the same group along this work, as most algorithms treat them with bias similarly and the challenges they face are related and will be referred as being simply rare chords.

Finally, there are some relevant points to be discussed regarding the relationship between some triads and their more complex counterparts. There are some complex chord qualities that are, by definition, composed of a more simple triad. Such chords have a very strong relation between them, as the more complex is a sort of "extension" of the other. An example can be seen between the seventh and major chords, as the seventh chord is built by including an extra pitch to the major triad. Besides, it is possible for different complex chords to share the same basic triad, as is the case with the major seventh chord, which is also based on the major triad. These relationships are essential in the context of ACR, as it is very common for biased classifiers to mistake the complex chords for their more common counterparts, as will be seen in future chapters. Besides this issue, another kind of ambiguity is intrinsic to the definition and naming of chords, which might generate further issues.

Chords with different quality and root names can be composed of the same pitches. Physically, when looking simply at the soundwaves being produced and the sound being heard, these two chords will be exactly the same. However, considering the role they play in music theory, because of reasons that are out of the scope of this work, these two chords can have completely different roles. The important point is that these enharmonic chords, as they are called, generate a certain degree of ambiguity and might create some further obstacles for classifiers. One example of such chords can be seen in Figure 2.3.

[2]Extracted from: <https://www.pianochord.org/>



(a) `Gm7` chord composed of G, Bb, D and F pitches

(b) `Bb6` chord composed of Bb D F and G pitches

Figure 2.3 – Example of enharmonic chords, in which differently named chords contain the same set of pitches [2]

2.2 Automatic Chord Recognition

This section details a bit of the more specific problems related to ACR, and how this task relates to others that are similar to it in Machine Learning. As already presented, the main goal in ACR is to identify a chord being played on a given audio excerpt. This raises several sub-problems, such as which kind of feature to use, how to define a chord, consistency over time and ambiguity (PAUWELS et al., 2019). For instance, it is possible to use the raw audio (time domain) as a feature for a classifier, or use the Fourier transform (frequency domain) to extract frequencies from it, or even further, use a Constant-Q Transform (CQT) (BROWN, 1991), which is much more suited to a musical environment because of its properties. For instance, one advantage it provides is that the bin sizes used in its transformation have a variable size, which allows them to be smaller in lower frequencies and larger in higher ones. Because of the nature of musical audio, in which pitch frequencies double every octave, this generates a more suitable transformation process, and better captures both extremes with a single operation. Defining solutions for these problems is not always immediate and allows for different branches of contributions to arise from them.

Atlhough ACR presents some particular characteristics, it presents common challenges with other Machine Learning problems. For example, image classification problems also have to deal with rare categories or imbalanced datasets, which might involve specific loss functions or training strategies. With that in mind, a new path of opportunities to be explored is created. On the other hand, it is important to account for some major differences between these fields, as the audio domain is intrinsically sequential, while image recognition is not. One example that might require specific treatment is that of data augmentation. Data augmentation means applying certain techniques to generate slightly modified or synthetic data samples,

in order to create a larger and more varied dataset. In the case of images for example certain visual augmentation may be applied slightly modifying the image, however in audio, these same techniques might not be available and some equivalent needs to be found. Regarding the generic audio domain, it is even more similar, as it handles the same data structure (i.e., audio signals), meaning that it is also sequential. However, it is interesting to consider that an extra layer of treatment to the musical data can be explored, as some specific augmentation steps or pre-processing techniques can be applied to music and might not be that useful for audio itself. For instance, pitch-shifting is an augmentation technique widely used for musical data (MCFEE; HUMPHREY; BELLO, 2015; HUMPHREY; BELLO, 2012; MCFEE; BELLO, 2017), as it augments the data in a predictable way for the labels to also be updated. Besides, as mentioned before, a transformation such as the CQT is much more useful when applied to musical data and not audio excerpts in general.

3 RELATED WORK

The work presented here will focus on improving the performance of rare chord recognition, and will be done by evaluating some classifier training techniques. One of these will also involve the generation of a weakly labeled dataset of timed chord annotations, which later on be presented as part of one of the proposed training techniques. In this section, an overview of the state of the art in the ACR community will be first provided. Next, the focus will shift more towards the more specific work in ACR related to this one, with rare chord recognition, chord class imbalance, and dataset generation.

Finally, a brief overview of techniques that attempt to tackle these same issues of class imbalance and few labeled datasets will be explored. These techniques, although applied to different domains, could be explored and an equivalent solution for the ACR domain defined.

3.1 Automatic Chord Recognition

Automatic chord recognition, as the name states, is a field of research that aims at automatically identifying and labeling musical chords in audio excerpts. With more than 20 years of history and development, chord classifiers had initially started as being much more knowledge-driven, such as the proposed solution by Fujishima (1999), in which a 12-dimension binary vector, known as chroma feature, was proposed and compared to their chord template representations in order to identify the chord being played. This work provided the base on which most of the early work in ACR was built upon (PAUWELS et al., 2019).

In more recent ACR approaches, these knowledge-driven classifiers have shifted towards a much more data-driven context, attempting to replace the framework of chroma features initially proposed (PAUWELS et al., 2019). Some early examples of this shift can be found in (SHEH; ELLIS, 2003), in which Gaussian models were used to represent the chroma features and Hidden Markov Models (HMM) to account for the temporal relation between subsequent chords. Lee (2007) also explored the usage of HMMs, but trained them to be genre-specific and on synthesized audio in order to avoid the massive human labor required for data annotation.

Continuing on this evolution towards data-driven classifiers, more recent

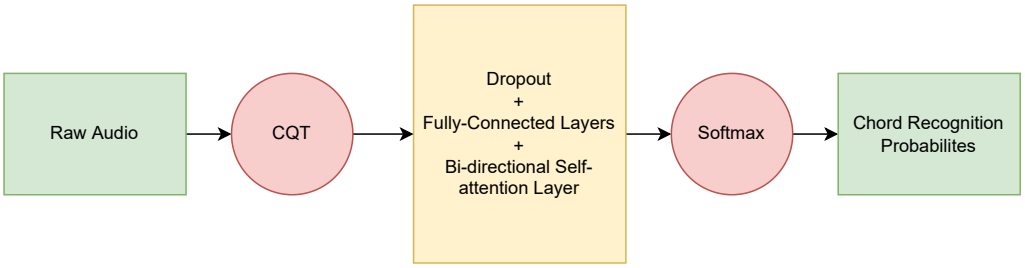


Figure 3.1 – Simplified version of architecture proposed by Park et al. (2019)

work has explored it even further with the usage of Deep Learning techniques. In (BOULANGER-LEWANDOWSKI; BENGIO; VINCENT, 2013), a Deep Neural Network (DNN) is explored in order to extract audio features and a Recurrent Neural Network (RNN) was used to provide the sequence of chord labels taking into consideration their temporal relationship. In (MCFEE; BELLO, 2017), an encoder-decoder architecture is used with a Convolutional Neural Network (CNN) and recurrent networks in the encoding step to convert audio to a latent feature, and afterward, in the decoding step, convert these features to chord labels.

One key concept of ACR is the need to provide a time-aware classification of the chord sequence. In the methods cited above, it was possible to see the tools used for this purpose, such as HMMs and RNNs. In more recent work, also following the line of more data-dependent solutions, the usage of Long Short-Term Memory RNNs, such as in (DENG; KWOK, 2017) and (Wu; Li, 2019a). The usage of Transformer architectures has also been explored in this context, such as in (PARK et al., 2019) and (CHEN; SU, 2021).

Park et al. (2019) explored the concept of attention-based models, which provides a more flexible way of using the input data in sequence translation problems. Their work uses a Bi-directional Transformer architecture in this context, allowing them to better capture long-term dependencies in the sequence. This last point is a very relevant aspect in the context of music, as it relies heavily on these long-term relations because of the chord relationships and harmony component in music. Their approach managed to achieve competitive results when compared to other state-of-the-art techniques in ACR and has been also recently explored as the ACR module for automatic lyrics transcription in (GAO; GUPTA; LI, 2022). A brief overview of the architecture proposed by Park et al. (2019) can be seen in Figure 3.1.

3.2 Class Imbalance in ACR

Class imbalance is an issue present in many different classification problems, and ACR is not an exception. Furthermore, with the current scenario in the ACR community, in which classifiers have moved from a knowledge-driven domain to a data-driven approach, the need for more and more data made this issue even more evident. New techniques and architectures have been proposed to handle this problem. For example, Wu and Li (2019b) use digital audio to train a CNN for feature extraction and (KORZENIOWSKI; WIDMER, 2016), which applies detuning to the input audio in order to augment its training dataset, applying it to a CNN for audio extraction followed by a Conditional Random Field (CRF) for decoding the final chord sequence. Although these techniques explore ways of minimizing these issues, the problem of class imbalance still remains open for this task.

Besides the need for different techniques to handle a data-driven problem, it is necessary to have the supporting data for such a scenario. The literature reviewed here presented three different kinds of strongly-labeled datasets used for training: manually labeled, synthetic and semi-automatic datasets. They are better described next.

The labeled datasets are the ones with a set of songs that were manually analyzed by musical experts, which defined the corresponding labels for each excerpt based on their own knowledge. Out of these datasets, it is necessary to outline three established examples that are widely adopted by the ACR community. First, there is the Billboard dataset (BURGOYNE; WILD; FUJINAGA, 2011), which is a selection of songs from the Billboard "Hot 100" chart with their labels hand-annotated by a group of experts. Next, there is the Isophonics dataset (HARTE et al., 2005), which is composed of hand-annotated songs from The Beatles, Queen, Zweieck, and Carole King. Its labels were also manually defined, but the validation was made through the continuous usage of them by members of the MIR community. Finally, the RWC Popular Music dataset (GOTO et al., 2002) provides 100 chord annotated songs.

The second group of datasets originates not from manually annotated labels on pre-existing songs, but generated labels for synthesized music. The idea behind it is that, as the music is synthesized, the labels are known beforehand, allowing the final output to be the synthesized audio and the precise chord annotations for it. One

of the main examples in this scenario is the MAPS dataset (EMIYA et al., 2010), where piano recordings were generated through the use of virtual piano software or Disklavier. The usage of synthesized datasets has already been attempted, however, the interest for having real music datasets still remained (BURGOYNE; WILD; FUJINAGA, 2011). Although the usage of synthetic data does help the training process, there are some specific components of real audio that are not so easily reproducible by synthetic data and might not allow it to completely replace it during model training and evaluation.

The third group presented in this discussion are the semi-automatically labeled datasets, which are a mix between real audio and synthesized data. Guitarset (XI et al., 2018) uses hexaphonics pickups to precisely detect onsets and offsets. IDSMT-SMT guitar (KEHLING et al., 2014) applies a multi-pitch detection based algorithm to estimate the musical content of the audio recordings. Here, the original audio is captured, but more precise labels are generated automatically.

Another important aspect that will be discussed in this work will be the use of chord labels from online communities. Therefore, it is important to present some related work that also explored online communities. In (MAUCH; FUJIHARA; GOTO, 2011), these labels were used to improve the performance of a lyric alignment algorithm, and Odekerken, Koops and Volk (2020) used them to estimate chord labels for one of the entries of an ensemble classifier.

In addition, it is possible to see some approaches that propose solutions for the imbalance issue through different training and classifier frameworks. In (Wu; Li, 2019a; MCFEE; BELLO, 2017; JIANG et al., 2019), we can see some approaches aiming to improve the classification of rarer classes through the use of specific frameworks targeting this problem. Rowe and Tzanetakis (2021) proposes both an architecture and the application of a specialized training technique targeting this challenge. One interesting point that will be made clearer later in this work is that most of the techniques above are compatible with the solution which will be proposed here, as it relies on an iterative training process, and each iteration could apply these techniques.

3.3 Class Imbalance in Related Fields

As mentioned before, there is a lot that can be drawn from other classification tasks to be used in ACR, especially when dealing with class imbalance, and a thorough review of class imbalance in CNNs can be seen in (BUDA; MAKI; MAZUROWSKI, 2018). In (BEERY et al., 2020), the classification results are improved through the use of synthetic images of rarely seen animals are used to improve animal recognition in images. In (LIN et al., 2017), the impact of the loss of easily classified samples during training was mitigated through the use of a modification to the cross-entropy loss (called focal loss). The use of noisy labels might be handled in different ways, such as by using an additional noisy layer (BEKKER; GOLDBERGER, 2016), by exploring knowledge distillation (LI et al., 2017; HINTON; VINYALS; DEAN, 2015), combining multi-source of features with filtering of noisy labels (Yadati et al., 2018), or by co-training two networks with noisy labels (HAN et al., 2018), to name a few.

Self-learning techniques have proven useful when working with noisy data. In (LI et al., 2019), a supervised teacher model is first trained with labeled data, and an enlarged training set is obtained based on its predictions of the weakly (or unlabeled) data. Xie et al. (2020) proposed the Noisy Student algorithm, in which a combination of labeled and unlabeled data was used with a self-learning technique to increase the overall accuracy in the context of image classification.

Our work follows the Noisy Student strategy by extending it to the musical domain and also exploring the automatic creation of a dataset with weak labels that are combined in the self-supervision process, as described in the following chapters. The translation of this technique to the audio domain is not trivial, as there are several sequential aspects of the domain which need to be taken into account when preparing the iterative self-learning process. Besides, we'll also explore the usage of weak labels in these same context of self-learning techniques in order to further improve its results

4 GENERATING A WEAKLY LABELED DATASET

This chapter presents a technique to generate a weakly labeled dataset for ACR. As discussed in the previous chapters, a very high cost is involved in obtaining labeled chord annotations. This is a result of the necessity of having expert annotators working towards labeling existing musical files or generating new audio and annotations from scratch. Therefore, it is very interesting for future algorithms to have access to an additional source of annotations, as this adds an extra variety of musical styles and allows for more recent songs to be included, despite the weak labels. The application of datasets with noisy labels has already been seen in Chapter 3, but related to image recognition tasks. Later in this work, we will present an approach for exploring noisy labels in chord datasets for chord recognition by adapting approaches used for images.

The algorithm proposed for the dataset generation is based on two different sources of information that can be obtained with less effort than annotated chord labels. The first source is retrieving chord information for any given song from one of the many online chord annotation communities, which associate the chord with the song's lyrics. The second source is the timed-lyric annotations for these same songs, which are not available by default, but requires no expert musical knowledge for labeling – hence, being easier to annotate. The next sections begin by detailing the data collection process for both data sources. After that, the pre-processing steps involved in preparing the data collected for the algorithm are detailed and, finally, the technique used to combine both sources of information into the weakly labeled dataset is presented. A detailed step-by-step of this process can be seen in Figure 4.1.

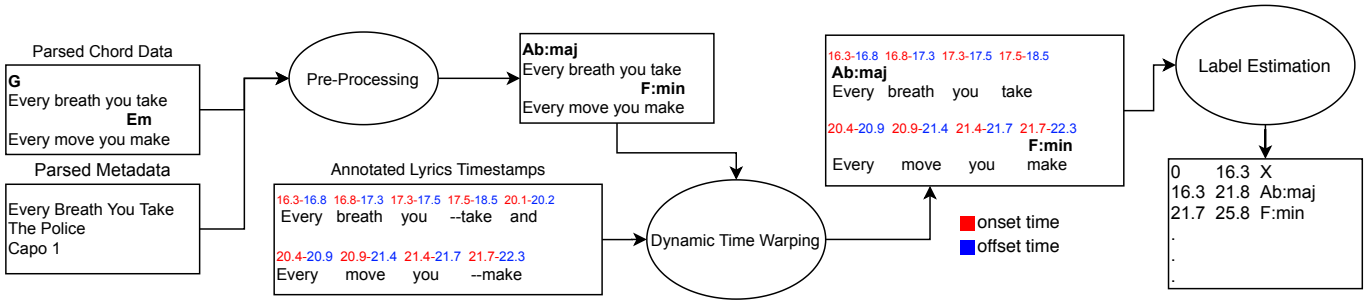


Figure 4.1 – Example of label generation process using an excerpt from the song Every Breath You Take

4.1 Data Collection

As mentioned before, the number of publicly available annotated chord datasets is limited, and creating new ones is not an easy task. As a consequence, the variety of songs and styles available for training a classifier is also limited. However, there is an almost limitless quantity of annotations associating the chords of a song with its lyrics. These chord-lyrics files can be found in online chord communities, where users create and review the annotations. Each song might receive multiple versions of the chord annotations, and the users themselves provide feedback regarding their quality. The proposed technique aims to explore this vast amount of data and provide a way of using it to contribute to improving automatic chord recognition classifiers.

As will be seen later on, this information by itself might not be as useful as expected because it provides no time information for the chord onsets and offsets. Therefore, a second complementary source of data will be required to include this information in the final annotation. First, we present the structure in which chord-lyrics files from online communities are usually found and how this will be useful for the dataset generation. Then, we provide a way to obtain the second part of the required data (i.e., the time-annotated lyrics).

4.1.1 Chord-Lyric Data

Chord-Lyrics files from all chord communities usually follow the same structure, in which the song lyrics are presented every other line with chords between them, right above the word – or part of the word – on which their onset is supposed to happen. However, these files might sometimes contain other information, such as section names of the song, some description regarding how to play, tabs for guitar solos, or even if capo[1] usage is required. Therefore, retrieving the association between chords and lyrics from these files is not trivial. Figure 4.2 shows an example of how one of these files might look.

It is important to notice that although there is other information present in

[1]Capo is a musical device often used in string instruments such as guitars. It is placed on the neck of the instrument, reducing the length of the string, thus raising the pitch and transposing the notes being played

these files, we can clearly infer a relationship between chords and the song lyrics, as mentioned by Mauch, Fujihara and Goto (2011). This relationship is the key aspect of the technique presented here, as it allows our algorithm to extract the exact word in a song during which a chord change is supposed to happen and also which chord should be played next. With this information in hand, it is then possible to estimate the chord sequence from a song and during which word the change should happen. However, there is still one significant gap between this data and an annotated chord dataset: time information. As mentioned before, an ACR dataset relies both on the chord label information and the time label for the onset and offset of each chord. A technique to obtain such data will be presented shortly, but first, a brief description of some other useful information found on such chord-lyric files will be detailed.

Besides providing the relationship between chords and lyrics, other useful data can be extracted from such online community chord files. First, it is possible to extract the capo information for the song, allowing our technique to adjust chord labels accordingly. Structural information about the song may also be extracted, and it might indicate that some sections might repeat themselves, usually also repeating chord sequences.

Finally, for the rating provided by users of online communities, the developed solution supported two different scenarios based on the different formats the ratings appeared along the chord communities. In the first scenario, we can retrieve the actual average rating provided by the users for each file, so that it represents the actual quality metric perceived by the users for a given chord annotation file. In the second scenario, however, no explicit rating is available. Instead, only the number of people who rated the file is available, and we consider this information as a source of positive feedback.

4.1.2 Timed-Lyric Data

As described above, it is necessary to obtain a way of linking chord data with timestamps in order to obtain an annotated ACR dataset. The technique proposed here relies on joining a data source that associates chords and lyrics with another source that associates lyrics and timestamps, therefore allowing for an association between chords and timestamps to be inferred. In this section, we describe the second source, which provides the lyric–timestamps relation. We assume that access

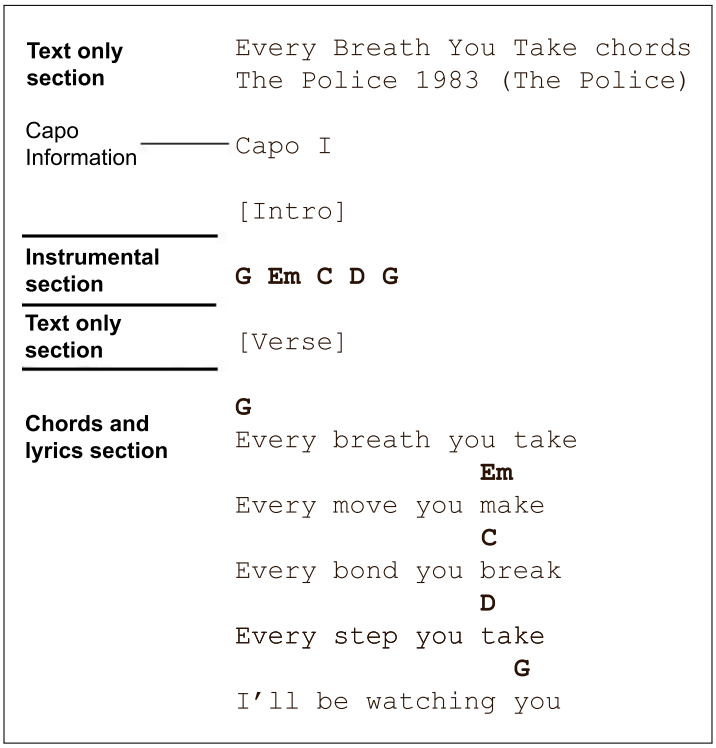


Figure 4.2 – Example of extracted sections from an excerpt of a crowdsourced chord label page

to these time-annotated lyrics is available[2].

In order to proceed with the dataset generation, we obtain lyric annotations from an existing dataset called DALI (MESEGUER-BROCAL; COHEN-HADRIA; PEETERS, 2018), which was initially intended for usage in singing-voice MIR tasks. It provides detailed lyrics onset and offset for over 5,000 songs with different styles, countries, and decades. This association between lyrics and timestamps is illustrated in Figure 4.1 (see Annotated Lyrics Timestamps). The DALI dataset also describes how to obtain the exact same audio files used to generate the annotations, maintaining the coherence between the audio and the timed onset and offset labels.

4.2 Pre-processing

Considering now that data from both of the sources above have been collected, it is possible to start describing which pre-processing steps are required to clean this data and prepare it for usage in our technique. First, let us consider the data obtained from online chord annotation communities, which provides a clear

[2]Even if it were necessary to annotate these lyric files, the task could be accomplished by a group of non-experts in musical transcription, requiring only knowledge of the language present in the lyrics and the effort to label each word.

relation between the lyric words and the song chords. Such information is often mixed with other not-so-relevant sections in the files obtained from the communities, as illustrated in Figure 4.2. As we can see, this file excerpt also includes some indications of the different kinds of sections that can be identified, such as sections with lyrics and chords, text sections, or instrumental-only sections. Although helpful information could be retrieved from text-only and instrumental-only sections, the technique proposed here for generating a dataset will be focused on using the relation between chords and lyrics, and therefore a crucial pre-processing step is to identify and extract only such sections from the obtained files. The identification of these sections is made based on the notation present in such files, which often represents them with a chord placed right above the text it is associated to. This way, it is possible to search for sections following this pattern and retrieve them, as seen in figure 4.2.

Besides identifying and extracting relevant sections, it is essential to consider that chord annotations are created manually by a large community of individuals, each one with its own preferences regarding chord notation. Thus, another important pre-processing step is to convert the chord notation to a standard format. The standardized notation that will be used is the one proposed by Harte et al. (2005), which has been widely used in the ACR community and is usually compatible with state-of-the-art classifiers. This notation consists of a way of defining the chord root note, quality, bass and other intervals contained in it. Some examples would be the C major chord (`C:maj`), or the A minor with C as bass (`A:min/3`, where 3 represents the pitch at the third degree of this chord, which is C).

Finally, there is also another important piece of information that can be extracted from these files, which is not related to the chords or lyrics sections. Usually, the files contain a certain degree of metadata available for collection and, if care is not taken, might even interfere with the resulting annotations. Most of the chord notation in these files was done by and for guitar players, which means that some files contain capo and tuning information explicitly made for the guitar. For instance, if a capo is indicated for usage on a certain song, all chord notations will be made relative to the position of the capo on the guitar. This means that the indicated chord is not the same as the one being played, and requires a pre-processing step to transpose it according to the capo position. An example of capo pre-processing can be seen in Figure 4.1, in which the chord notation was updated

based on the presence of a Capo in the first fret position, transposing the chord labels up by one semitone. The same logic applies to songs that indicate that a different guitar tuning is to be used. In this scenario, the chord label also refers to their position on the guitar and not the sound they produce, as the guitar's tuning is not the standard one. This also means that these chords need to be adjusted, so the label actually represents the correct sound that is expected to be played.

4.3 Generating the Label Files

After pre-processing and converting the chord annotations to a final standard format and filtering the lyric annotation files, the next few steps will be related to label generation. The description of this process will be divided into two parts: merging information from both files into a single one through an adapted Dynamic Time-Warping technique and estimating the final ACR annotation files, which are composed of the predicted chord and its onset and offset.

4.3.1 Dynamic Time-Warping

One of the crucial steps in attempting to join the two files is to handle the small differences present in them, especially on the word level. Both files were generated from independent sources, one being from online chord communities, and the next coming from an already existing lyrics dataset. Therefore, words can be written differently, being them typos, different word separation, or sections and words that were simply not present in one of the versions. Because of these differences, it was not possible to simply expect an exact match for both files from the same song, and a certain degree of uncertainty had to be dealt with.

The selected technique to handle such issues was Dynamic Time-Warping (DTW) (MÜLLER, 2016), which is a method commonly used with the purpose of aligning two time series but can be used with other types of data that present sequential information. The main idea behind it is that a measure of distance is calculated between the data points of each series, and DTW considers this distance as a cost function in order to find the lowest cost path (i.e., the alignment of both series) between the two files. Let us consider two different time series $\boldsymbol{X}$ and $\boldsymbol{Y}$,

each one being a vector of datapoints $x \in \boldsymbol{X}$ and $y \in \boldsymbol{Y}$. The main idea behind the DTW is to calculate the distance between these two vectors and, by using the smallest obtained distance, align these two vectors, considering that their data points are ordered in time. For this purpose, a cost function between each pair of datapoints (x, y) is defined as $d(x, y)$, and used in the calculation for the final alignment.

In our scenario, the time series are actually lyrics files, and the words from each file as the data points. To adapt the DTW for matching text files, we need to define a cost function and use it to obtain the lowest-cost alignment between the chord and lyric annotation files. As the objective of this alignment is to take typos, misspellings, and missing sections into account, the most appropriate cost function for the DTW should take the degree of difference of the words into account when calculating the cost. A widely known metric for this purpose is Levenshtein's string edit distance (LEVENSHTEIN, 1966), which calculates the number of character changes required in order to change a given word into another. This metric will, in this scenario, provide low costs for simple typos and misspellings, and the DTW itself would be in charge of ignoring missing sections, as the cost for them in the comparison would be very high. Hence, our cost function $d(x, y)$ is defined as

$$d(x, y) = lev(x, y), \tag{4.1}$$

where $lev(x, y)$ is the Levenshtein's edit distance, and x and y are elements from our $\boldsymbol{X}$ and $\boldsymbol{Y}$ vectors, which in our scenario would be vectors of word tokens extracted from our data sources. An example of how this alignment would be applied can be seen iNn Figure 4.1, where our word tokens would be extracted from each of the sources serving as input to the "Dynamic Time Warping" step, and their resulting alignment can be then extrapolated to retrieve the original positions of both the token timestamps and the chord labels, as seen in the output example for the step.

In more detail, an example of the matching process can be seen in Figure 4.3, in which two different lyrics sources are passed to the DTW function, using the distance measurement described above. Its output provides a word-by-word match between these two sources, accounting for extra words, repetition, and spelling mistakes. The third column in the output provides the resulting distance between the matches, which the DTW aims to minimize.

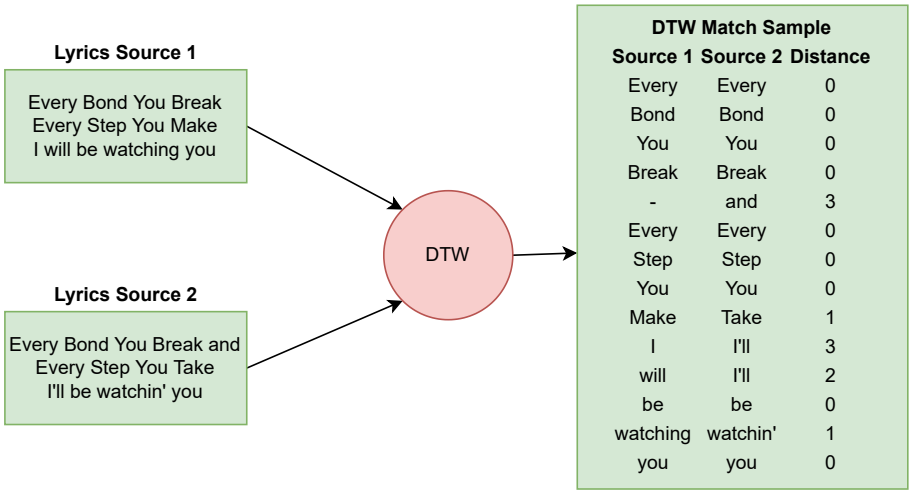


Figure 4.3 – Sample of how the DTW is applied to match two different text sources, with the resulting distances between matches.

4.3.2 Label Estimation

The procedure described so far produces an association between lyric words, their onsets and offsets, and the position of the chords related to these lyric words. The final step is to generate the final chord and timestamp association. As seen in the data collection and pre-processing steps, the chord is associated with a particular word in the online chord community files by being placed above it (check the "Chords and Lyrics" section in Figure 4.2). Thus, it is possible to infer the chord onset timestamp from the time information obtained for the word placed closest to it on the line below. Also, its offset can be determined based on the next chord onset, as the chords usually follow one another. In case an instrumental- or text-only section begins, it is also possible to consider that the previous chord should end there, as our technique cannot infer information without both lyric and chord components. Besides, in order to take these sections into account and leave no gaps in the song timeline, the proposed generation technique will mark these sections with a label "X" from their onset to their offset, which indicates that no label could be obtained for these parts. The output file follows the standard format used for ACR dataset notation, where each line provides three columns with the onset, offset and chord label, respectively, and thus providing a chord and timestamp association required for training classifiers. The rightmost box in Figure 4.1 illustrates an example of final time-stamped chord annotation.

The approach described in this chapter can generate a weakly-labeled ACR dataset. Since it does not involve musical experts, it presents a large potential for

increasing the variety of songs used in the process of training classifiers and allows us to explore new techniques for self-training using weak labels. In the chapters that follow, this dataset will be used to improve the performance of a baseline chord recognition approach, particularly for rare chords.

5 IMPROVING RARE CHORD RECOGNITION

In this chapter, we present the techniques applied to improve rare chord recognition. We first introduce a technique that only uses unlabeled data, and then our final approach, in which weak labels are also included for training the model. The techniques presented here aim to tackle one major problem in the ACR task, which is the difficulty in obtaining good predictions on rare chords, while also attempting to maintain the overall prediction quality. The experiments were performed with each of the proposed changes individually and by combining the most promising ones to verify if their combination could boost the results even further. This will reflect directly upon the experimentation setup presented next.

The following sections of this chapter continue with a description of the two different approaches. The first one is based on an iterative training technique and a loss adjustment factor, both previously seen in image recognition tasks and that require no labels at all. The second method applies an extra step of confidence boosting based on the weakly labeled dataset described in Chapter 4 to the iterative technique presented in the first experiment.

5.1 Improving Recognition With Unlabeled Data

This section will describe the two main directions we explored to improve the classification accuracy, particularly of rare chords. The first direction described will be the usage of a focal loss (LIN et al., 2017), adapted to the context of sequential data. The second direction will be the extension of an image recognition self-learning training method, known as Noisy Student (XIE et al., 2020), to a sequential data context, which is the case of audio and musical data in particular.

5.1.1 Focal Loss

One of the main issues this work aims to tackle is the strong imbalance in ACR datasets. Besides the imbalance in the duration of chords, there is also an imbalance in chord complexity. As mentioned before, some chords contain more notes than others, an unusual combination of notes, or even certain combinations of notes

that make similar-sounding chords have different labels. It is in this context that the focal loss technique is expected to provide an overall performance improvement, but in particular for these more complex chords.

Most classifiers used in ACR rely on a standard log-loss function for their training process. This function assigns a certain value based on comparing the label prediction confidence and the actual label for the excerpt. In this scenario, even predictions that provide high confidence for the correct label can generate a small impact on the loss function if the number of samples is small. Let us consider datasets comprised mostly of a certain group of chord types, which additionally are relatively "easy" to predict. These circumstances might end up generating small individual loss components for each of these common chord instances since correct and high-confidence predictions might be made. Although the loss components for these easy chord samples tend to get closer to zero, because they are comprise most of the dataset, the sum of these small components, results in a large value. Meanwhile, rare chords generate a large loss component each, but because they are rare throughout the dataset, the sum of their loss components might not be significant enough when compared to the overall loss of more common chords. This scenario could make these rare, complex chords even more difficult to be learned by the classifier.

In order to reduce the impact of these easy and high-confidence predictions, the focal loss presents a modified version of the standard cross-entropy loss function that reduces the value for easier, already-learned, training samples. This is done through the introduction of a power term in the traditional log-loss function, given by

$$FL = (1 - p_t)^{\gamma} \log(p_t), \tag{5.1}$$

where p_t is the estimated probability for a ground truth chord and γ is the power term mentioned above, and can be adapted to be influenced by how much the weight of the already learned samples will be reduced during training. The higher the value of γ, the less influence easier samples will have on the summed loss function for multiple samples, and the loss should then "focus" on harder chords.

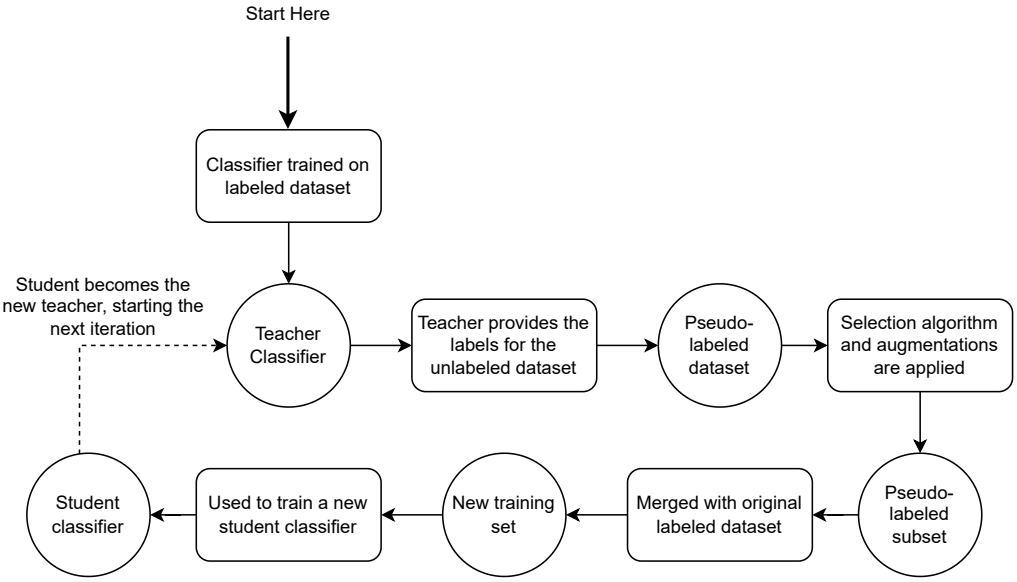


Figure 5.1 – Noisy Student Iterative Process

5.1.2 Extended Noisy Student

The second direction explored involved extending a state-of-the-art self-learning technique from the domain of image recognition to the sequential domain of audio. The technique known as Noisy Student (XIE et al., 2020) relies on using a large unlabeled dataset, along with a small labeled dataset using an iterative, teacher-student training process.

The process starts in the first iteration with the training of the classifier using the fully labeled dataset, generating the first teacher. This teacher is, in turn, used to classify the large unlabeled dataset, generating pseudo-labels for it. A selection algorithm is applied to these pseudo-labels in order to create a subset of high-confidence predictions out of them, with a similar amount of samples as the labeled dataset. Augmentation techniques are applied to this subset to make the pseudo labels samples become noisy.

Now, both the labeled dataset and subset of noisy pseudo-labels are merged into a new training dataset, which is, in turn used to train a student classifier. The student classifier then becomes the new teacher classifier, generating new pseudo-labels for the large unlabeled dataset and repeating this iterative process. A diagram with the definition of the iterative process can be seen in Figure 5.1.

Having an outline of the process done by this technique, it is now possible to dive further into some details for its execution, in particular, the subset selection and augmentation methods. The selection algorithm used in the original Noisy Student

algorithm relies on selecting, with replacement, a random set of samples out of the pseudo-labeled dataset that present prediction confidence above a certain, predefined, threshold. An important point here is that the number of samples selected from each of the different classes in the problem is the same, thus making the selected subset have a certain class balance with high-confidence predictions.

The second part that can be detailed is the set of augmentation techniques being used, as this will be one of the points that will need to be considered when extending the technique to the audio domain. Two different kinds of noise are used in the original technique: dropout (SRIVASTAVA et al., 2014) and stochastic depth (HUANG et al., 2016) during the training process, and input noise, through the usage of data augmentation techniques found in RandAugment (CUBUK et al., 2020), done directly on the selected subset of pseudo-labeled samples.

5.1.2.1 Adapting to the Audio Domain

Many of the strategies described above are not immediately compatible with the audio domain, which differs from the image domain mainly in the aspect of the former being sequential while the latter is not. Because of this crucial difference, it is important for some characteristics of the original technique to be reconsidered and adapted in order for them to work properly in this new context.

First, let us consider the pseudo-label selection algorithm and the issues that might arise from this change of context. The original technique relies on classifying single examples and providing a label with confidence for each image. This, however, is not true when dealing with audio, as each chord is linked sequentially to its neighbors, and there is a certain reliance on this context when making a prediction and assigning it a confidence value. Another issue that also originates from the need for context in ACR datasets is that a truly balanced dataset can never be obtained: as all samples must be selected along with their context, they will most likely include different labels along with the chosen one. Finally, as we have already seen, most of the samples in an ACR dataset are composed of major and minor chords, which means that it is very likely that many of the chords in selected subsets' contexts will be one of these chord types.

In order to address these issues, the following adaptations were proposed for the subset selection algorithm. To mitigate the requirement for context in sequential data, a minimum length of each selected excerpt has to be defined, here called

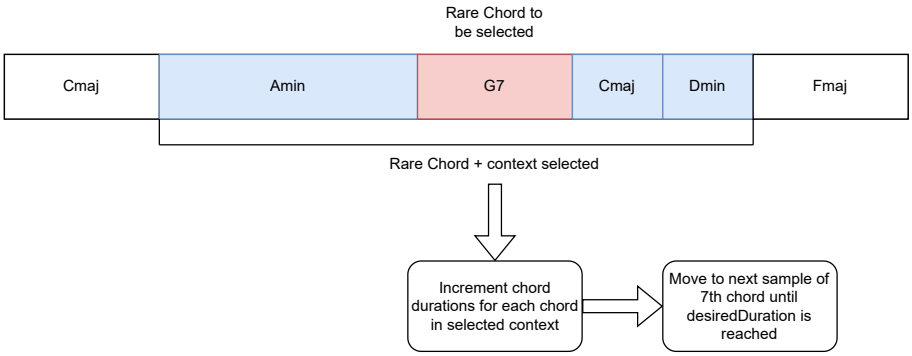


Figure 5.2 – Chord Selection Iterative Process

minLength. As the dataset composition is calculated based on the overall duration in seconds for a given chord type and not the number of samples, a definition of the duration for each chord class required to achieve a balanced outcome is needed and will be called in this document as desiredDuration. This will indicate the amount of time we would like to assign for each chord type selected, resulting in a balanced dataset. As the objective is to obtain a dataset of pseudo-labels with roughly the same duration as the original labeled dataset, the chosen desiredDuration is, by definition, the total duration of the labeled dataset divided by the number of rare classes present in the pseudo-labels. It is important to notice that only rare chords will be used in the selection, as we expect that the more common chords will be present in this final dataset just by appearing in the context around the selected samples. By selecting these more common chords directly, the resulting subset would likely be very imbalanced, going against the purpose of this technique.

The adapted technique, summarized in Algorithm 1, uses the variables defined above and selects the pseudo-labels with the highest confidence for each of the rare chord classes, with excerpts with at least minLength. This is done by iterating through rare chord class one at a time, and ordering all samples from that class based on their prediction's confidence. For each of the ordered entries, an audio excerpt centered on the sample is selected, which has a duration of at least minLength if the size of the sample is smaller than it, and adjusting the boundaries if the sample is at the beginning or end of the audio file. The selected sample is included in the pseudo-label dataset along with its context until the total duration of that chord type, here called selectedDuration, reaches the desiredDuration. When this is matched for a given chord type, the next one is selected, and the process continues.

An example of such iterative process can be seen in Figure 5.2, where we can see a musical excerpt being currently evaluated. It contains a G7 chord (in red),

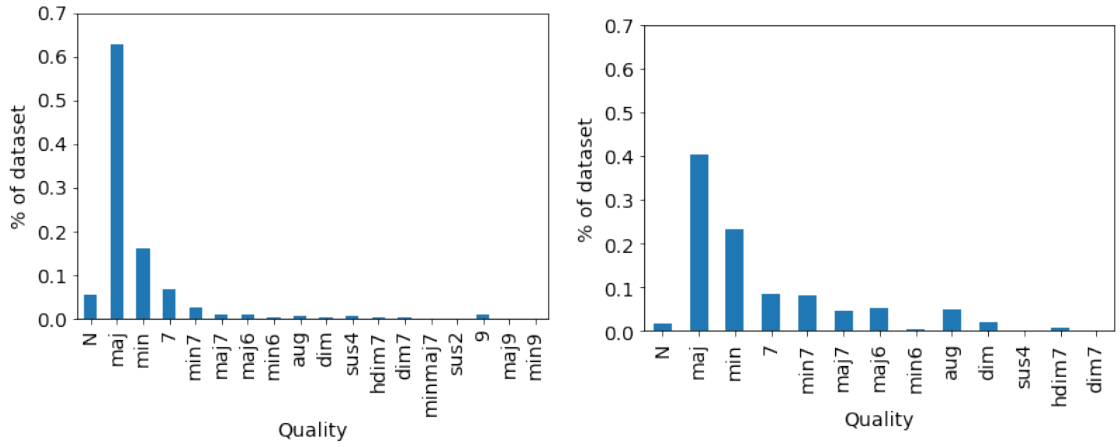
which is the rare class currently being evaluated (seventh chords), but also shows the associated musical context around it (in blue). Once such as excerpt of `minLength` is selected, the `selectedDuration` for each of these chords is increased. Then, if the `desiredDuration` for the current class being analyzed (seventh chords in this scenario) is reached, the next rare chord class excerpts undergo this same process. Otherwise, the next seventh chord excerpt is evaluated and selected.

The outcome of this adaptation generates a pseudo-labeled dataset that is much more balanced than a standard ACR dataset. This can be seen in Figure 5.3b, which presents the distribution of one of the pseudo-labeled datasets generated by the adaptation presented above, which can be compared with the Isophonics dataset distribution, presented in Figure 5.3a. As was mentioned in this section, the major and minor chords were still expected to dominate the dataset, as most of the chords in the context around the rare chords selected would be of that type. Although none of them were selected directly, they still ended up composing a large percentage of the dataset. Another important point to notice is that the set of chord types that the classifier we used could predict was limited, which results in a dataset without some of the chord classes present in the labeled dataset.

Algorithm 1 Balanced Pseudolabel Dataset Selection

```
minLength = minimum length for each audio excerpt
desiredDuration = total duration desired for each rare chord type
for each rare chord type ct do
    Filter labeled entries by ct ordering by prediction confidence
    Initialize selectedDuration = 0
    for each ordered excerpt do
        Create new excerpt centered on excerpt, with duration minLength
        Add it to the dataset, merging overlaps to avoid duplicates
        Increment selectedDuration by duration of the excerpt
        If selectedDuration ≥ desiredDuration then move to next chord type
```

The second point we considered in our adaptation was adding noise to the training and augmentation processes. The model noise techniques (i.e., dropout and stochastic depth) present in the original algorithm did not rely on any specific aspects related to image or audio and, therefore, could be maintained as-is. However, augmentation strategies applied directly to training data had to be adapted. Considering the context of MIR, some augmentation techniques can be used, such as pitch shifting or adding overlapping random noises. The classifier used for the training process already relied on pitch-shifting augmentation, and therefore another kind of random noise was also included for further augmentation. The usage of random



(a) Isophonics

(b) Balanced Pseudolabeled

Figure 5.3 – Comparison of chord type distributions on Isophonics and the Balanced Pseudolabeled datasets

audio excerpts from the UrbanSound8K dataset (SALAMON; JACOBY; BELLO, 2014) was used to include a certain degree of random noise in the pseudo-labeled dataset, which was done using the MUDA library (MCFEE; HUMPHREY; BELLO, 2015).

5.1.3 First Experiment Setup

The first experiment consisted of comparing both techniques described above, and a combination of them against a baseline classifier. A state-of-the-art ACR classifier architecture (PARK et al., 2019) was chosen and trained only using the Isophonics (HARTE, 2010) dataset, which contains 199 songs by The Beatles and Queen, in order to be used as the baseline. The chosen classifier works by receiving the raw audio as input data and generating a chord prediction for every time frame. This frame is the result of a windowing function applied to the raw audio, which selects short audio durations in time, and is able to generate a prediction for each. These predictions are condensed into larger units, which are in turn aggregated and turned into an onset/offset pair in time, along with the associated label for it, following the format proposed in (HARTE et al., 2005). In the case of the Noisy Student algorithm, which requires the prediction confidence to be included, the original architecture was modified to include the prediction confidence, calculated based on the softmax output of the last layer in the network. Regarding other training parameters (e.g., number of epochs and mini-batch size), we used the recommended

values provided by Park et al. (2019) in the original paper.

The first experiment being proposed here can be split into three phases, starting with the focal loss, followed by the extended Noisy Student by itself, and finishing with the combination of the best results for each of the techniques when tested alone. For the focal loss, different values of γ were tested to define which one would benefit the technique the most in this scenario. In the case of the extended Noisy Student, two different scenarios were created: one without adding the random noise augmentation as described above, and another one including it. Also, the iterations were performed until the results started to decline, which happened at the fourth iteration. After performing these tests individually, the best performing γ value for the focal loss and the best performing scenario for the extended Noisy Student were selected. With this information, the final phase of the experiment with the combination of these best scenarios for each of the first two phases was done. This phased experiment was required in order to reduce the required amount of time to execute the complete experiment, as the training and evaluation processes were very costly time-wise.

Three datasets were used in the different experiment configurations presented here. In the case of the focal loss, a single labeled dataset is required for training, similar to what happens in the case of the baseline classifier, and the same Isophonics dataset was used in this scenario. The Isophonics dataset was split into training and validation datasets, both here and in the baseline, with 80% for training and 20% for evaluation, used in the early stopping of the classifier.

Regarding the experiments involving the extended Noisy Student technique, a large additional unlabeled dataset is required for the iterative process. We used the DALI dataset (MESEGUER-BROCAL; COHEN-HADRIA; PEETERS, 2018), which is a singing voice dataset that contains over 5,000 songs in its catalog. Finally, another ACR dataset containing 100 songs, known as RWC (GOTO et al., 2002), was used as the test set for all scenarios to provide a cross-dataset evaluation for the techniques being tested and the baseline.

5.2 Improving Recognition With Weak Labels

After considering the scenario proposed for the techniques using only unlabeled data, this section will define a new extension to the modified Noisy Student

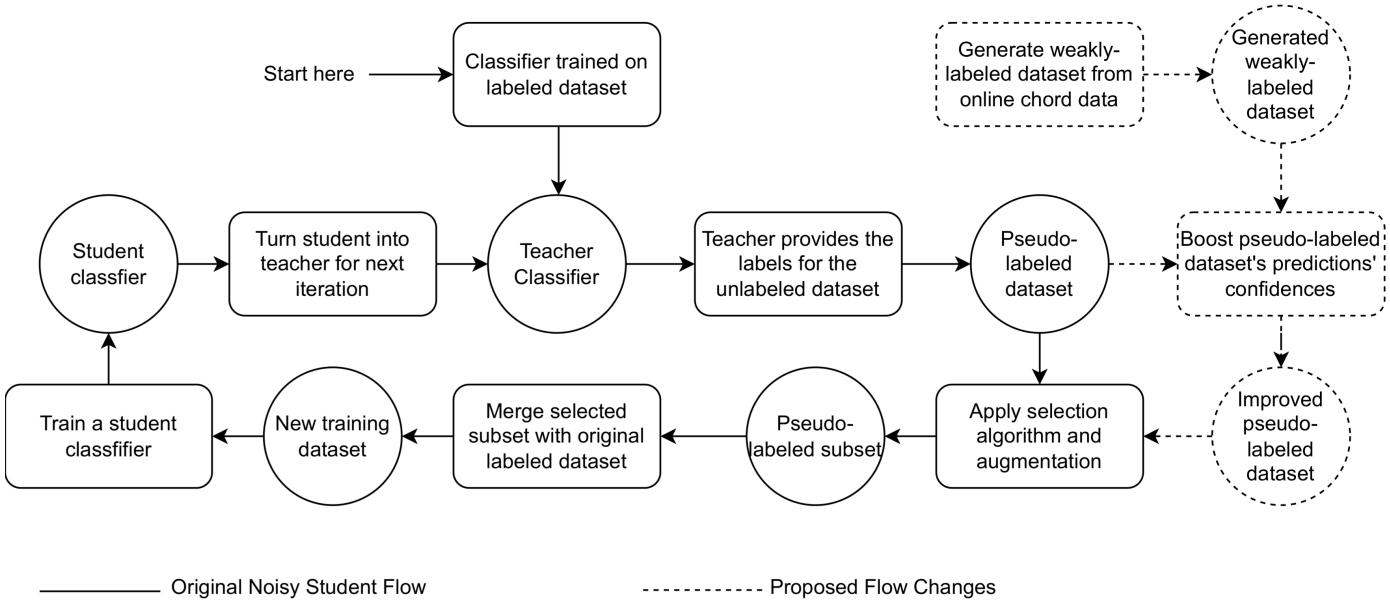


Figure 5.4 – Changes to the Initial Noisy Student Iterative Flow

technique defined above. This extension will rely on an independent source of weak labels to be used during the selection of the subset of pseudo-labeled predictions as a confidence boost whenever the classifier's prediction and the independent data source match their predictions. The main idea behind this extension is that by using two different, independent, sources of pseudo-labels, besides the one generated from the classifier's predictions themselves, it might be possible to improve the pseudo-label dataset selection process, generating better results.

The approach described in Section 5.1 requires only a set of Predicted Labels (PL), which are extracted from the teacher model. However, we believe that better results can be achieved by also including the External Labels (EL) provided in the weakly labeled dataset. Both sources can be considered somewhat independent, as one relies on the output prediction of the trained classified and the other originates from an external source with no relation to this same classifier.

Following the usage of these two datasets, this method aims to artificially increase the confidence of the teacher prediction used in the Noisy Student's pseudo-label dataset selection process whenever the labels from both sources match based on certain domain-specific comparisons. These comparison techniques must take into account some of the specifics of the ACR domain, such as the notes that compose chords and some chord similarity metrics. An overview of the changes to the iterative flow can be seen in Figure 5.4. In the following section, we will describe the techniques that were applied when testing this modification.

5.2.1 Calculating Prediction Confidence Boost

This section will present the three confidence-boosting techniques that will be tested along with the technique described here. Each of the techniques will rely on comparing two instances of chords, one from each of the two different label sources, the PL and the EL. Every comparison will result in a similarity metric between the two instances, which will be the value applied in the confidence-boosting formula. The following notation will be used: $L_p(x) \in \text{PL}$ is the predicted label of the teacher for a chord x, and $L_d(x) \in \text{EL}$ the corresponding annotation from an external source. For the original prediction confidence generated by classifier, the notation used will be $\kappa_p(x) \in [0, 1]$. Finally, using the similarity factor between the two chords instances as $s(L_p(x), L_d(x)) \in [0, 1]$, the following equation can be used to boost the original confidence, generating a new confidence $\kappa_o(x)$, as such:

$$\kappa_o(x) = \kappa_p(x) + (1 - \kappa_p(x))s\left(L_p(x), L_d(x)\right). \tag{5.2}$$

Note that Eq. (5.2) will boost the original confidence closer to its maximum possible value proportionally to the similarity between the two chord instances being compared. This way, it is possible to keep the same confidence format used previously, as its boundaries are still being respected, but allowing this similarity to be taken into account.

The first of the three similarity metrics that will be presented is also the simplest one. It involves simply checking if the label of the two chord instances is the same. This method does not take into account the many nuances of ACR and might miss out on some of the chords that are similar but have different labels, as we mentioned before. However, it provides a good starting point for a similarity metric. This similarity metric can be defined as

$$s_{eq}(L_p, L_d) = \delta(L_p - L_d) = \begin{cases} 1 & \text{if} \quad L_p = L_d \\ 0 & \text{if} \quad L_p \neq L_d \end{cases}, \tag{5.3}$$

where δ is the discrete unit impulse function.

The second technique considers some musical concepts to provide a more sophisticated similarity metric. Here, the chord label L is converted into its chroma feature representation, which consists of a 12-position binary vector $\boldsymbol{L}$ with each

position indicating the presence of the 12 different notes when collapsed into a single octave. The chroma feature vectors are generated for the two chords being compared, and the Chroma Feature Similarity (CFS) score is calculated for them based on the cosine similarity, as seen in (MÜLLER, 2016):

$$s_c(\boldsymbol{L}_p, \boldsymbol{L}_d) = \frac{\langle \boldsymbol{L}_p, \boldsymbol{L}_d \rangle}{||\boldsymbol{L}_p|| \cdot ||\boldsymbol{L}_d||}, \tag{5.4}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product operator.

This second technique takes into account a more specific aspect of chord similarity, which is the note components of a chord. However, there are still more complex aspects and ambiguities involved in chord comparison, which might make, for instance, similarly composed chords perceived as very different by the listeners. There have been studies in which such ambiguities have been dealt with through custom chroma templates that provide encodings for the harmonic components of fundamental frequencies belonging to a chord, such as the work by Oudre, Grenier and Févotte (2009). Besides, one very important aspect of chord comparison is the perception of the sound, and in this direction, there have been some studies analyzing music perception and attempting to provide some degree of mathematical modeling to reproduce such perception nuances (LERDAHL, 1988; CHEW, 2002). Also, in (HAAS et al., 2010; BERNARDES et al., 2016; MARQUES, 2019), some alternative distance estimations between chords were pursued based on these same techniques cited above.

Based on these aspects, the third similarity metric used will attempt to provide a more perceptual distance metric, which in this case was the Tonal Interval Space (BERNARDES et al., 2016). This technique relies on mapping the previously presented chroma feature vector into a complex-valued vector, the Tonal Interval Vector (TIV). This mapping is achieved through the Discrete Fourier Transform with a set of custom weights based on these mathematical models related explicitly to musical features and perception. This provides the tools to convert a chroma feature vector $\boldsymbol{L}$ into a more meaningful vector $\boldsymbol{T}$ from the musical point of view. Bernardes et al. (2016) also suggest using euclidean or cosine distances to calculate the similarity between two TIVs. In this work, the cosine similarity will be used because its range of possible values is known. The proposed metric is the same as

the one before, but with the TIV vectors instead:

$$s_{tiv}(\boldsymbol{T}_p, \boldsymbol{T}_d) = \frac{\langle \boldsymbol{T}_p, \boldsymbol{T}_d \rangle}{||\boldsymbol{T}_p|| \cdot ||\boldsymbol{T}_d||}, \tag{5.5}$$

where $\boldsymbol{T}_p$ and $\boldsymbol{T}_d$ are the TIVs from the chords in PL and EL, respectively.

5.2.2 Second Experiment Setup

The second experiment proposed here will be focused on evaluating the extension to the Noisy Student presented in the first experiment by including the confidence-boosting techniques. All three techniques will be compared to the baseline Noisy Student classifier, to the baseline classifier without Noisy Student trained both on a set containing only the labeled data and on another containing both labeled and weakly labeled data. The configuration will be very similar to what was done in the first Noisy Student experiment (i.e., the same classifier containing the prediction confidence output and the same hyperparameters).

Another significant difference from the previous experiment is that this one relies on the existence of an external source of labels. The source used for this weakly labeled dataset was already described in Chapter 4, with the generation of a dataset using the chord labels obtained from online communities. That way, the same large unlabeled dataset already being used for the training process will now possess an extra set of labels generated by the previously presented technique.

One crucial factor to consider is that the weakly labeled generated dataset has no clear way of being validated, as such a task would require extensive work by experts in musical knowledge. This experiment also aims to implicitly validate the usefulness of such a dataset by verifying if it can indeed improve the accuracy of the baseline ACR algorithm when compared to the Noisy Student baseline without the labels. This will be a comparison of the impact of using this dataset, along with the impact of the technique itself.

6 RESULTS AND DISCUSSION

The previous chapter presented an overview of the experiment setup, detailing two main experiments. In this chapter, a quick overview of the evaluation metrics will be provided, followed by an outline of the results obtained from each experiment. Finally, these results will be thoroughly discussed, with the positive aspects of the proposed techniques and the points identified that could be used for future advances.

6.1 Evaluation Metrics

One central point to be considered when evaluating ACR performance is that the metrics themselves might bias the analysis of our results if not treated carefully. One widely used metric is the Weighted Chord Symbol Recall (WCSR) (HARTE, 2010). It can be defined by first computing the Chord Symbol Recall (CSR) for each music track, defined as

$$\text{CSR} = \frac{|S_{pred} \cap S_{true}|}{|S_{true}|}, \tag{6.1}$$

where S_{pred} and S_{true} are the predicted and ground-truth labels, respectively. The "Weighted" part of the WCSR is obtained by weighing the CSR by the length of the track, yielding

$$\text{WCSR} = \frac{\sum T_i \text{CSR}_i}{\sum T_i}, \tag{6.2}$$

where T_i is the duration and CSR_i the Chord Symbol Recall of the i-th track, respectively.

Although this metric is widely used and provides a good overall score for the performance of a classifier, there is one point in which it lacks. As it does not take into account the classifier performance per chord type in any way, the chords that represent the majority of components on the datasets will have a much larger influence over the final results. Let us take, for instance, the major chords, which represent more than half of most datasets available: by focusing on a good performance on these chord types, the classifier will already obtain a good metric result even if it fails for less frequent chords.

As the purpose of this work is to also take into account the performance of

rarer chord classes, we will also explore metrics that consider potentially imbalanced datasets. The Average Chord Quality Accuracy (ACQA) is calculated similarly to the WCSR, with one main difference in that it weighs the performance of each chord class equally. It is defined as the sum of the WCSR for each chord type C (WCSR_C), divided by the number of chord classes, as follows:

$$\text{WCSR}_C = \frac{\sum_i C_i \text{CSR}_{C_i}}{\sum_i C_i}, \tag{6.3}$$

where C_i is the duration of the $i-$th chord instance of type C and CSR_{C_i} is the respective chord symbol recall. Finally, we obtain the ACQA score through

$$\text{ACQA} = \frac{\sum_C \text{WCSR}_C}{|\mathcal{C}|}, \tag{6.4}$$

with $\mathcal{C}$ being the set of chord types.

6.2 Results of the First Experiment

As described in the previous chapter, the first experiment consisted of comparing two different strategies to improve the accuracy of rare chords: focal loss and the Noisy Student. Table 6.1 presents a sensitivity analysis of parameter γ in the focal loss and also an ablation study that includes or suppresses different components. It is important to note that the noisy student technique ran for three iterations, as they started to decline in performance after that, but only the best results for each configuration are displayed in this table.

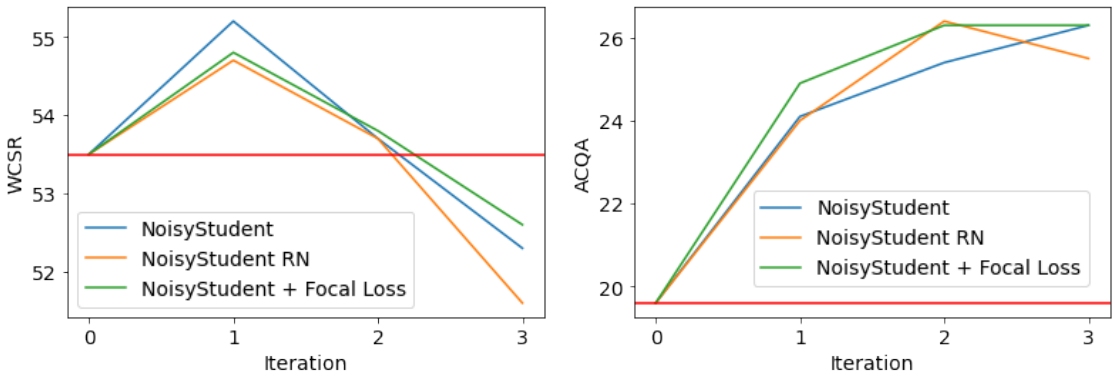
	WCSR	ACQA
Baseline	53.5	19.6
Focal loss ($\gamma = 2$)	53.5	21.9
NoisyStudent without random noise	53.7	25.4
NoisyStudent with random noise	53.7	26.4
NoisyStudent + Focal loss ($\gamma = 2$)	53.8	26.3

Table 6.1 – Weighted Chord Symbol Recall (WCSR) and Average Chord Quality Accuracy (ACQA) for each of the setups in the first experiment performed.

It is possible to see in Table 6.1 that all the experiments provided an improvement over the baseline in the ACQA metric while keeping the WCSR very similar to the original value. Out of the setups provided, the Noisy Student component seems

to have had the largest impact, with the best ACQA metric in the Noisy Student with random noise, while the largest WCSR was found when Noisy Student was combined with the Focal Loss.

As mentioned before, the NoisyStudent ran for three iterations, so the plots presented in Figures 6.1a and 6.1b provide the scores along each iteration for the WCSR and ACQA metrics respectively. Here, some interesting aspects can be observed. For instance, all techniques initially provided a significant gain over the baseline for the WCSR, which later started to decrease. Meanwhile, the ACQA score continued to increase for almost all setups along the iterations. One possible explanation for this behavior could be that the classifiers were starting to get better at identifying the rare chords, while their performance for the more common chords was decreasing, indicating slight overfitting towards these rarer chords. Another important factor to consider here is that such compromises had already been observed in (CHO, 2014), where the experiment performed was unable to improve both scores at the same time. In the scenario presented here, we were able to improve both metrics, as seen for the first two iterations in all setups. The techniques provided improvements on both metrics at the same time, and the results selected as the best ones were the ones with the highest ACQA (in iteration 2), which were then compiled in the previously presented Table 6.1, with the Noisy Student with random noise being the one with the highest ACQA overall score.



(a) WCSR performance (b) ACQA performance

Figure 6.1 – Performance over the Label Equality, Chroma Feature Similarity, TIV Similarity, compared to the Noisy Student (blue) and classifier (red) baselines during the first experiment

For the sake of comparison, Cho (2014) reported a gain of 20% in ACQA versus a reduction of 3% in WCSR. Meanwhile, the approach proposed here in its best scenario obtained an increase in both metrics, with 0.4% improvement in

the WCSR and 34.7% in the ACQA score. Considering the WCSR metric, the scenario that provided the largest increase happened during the first iteration of the NoisyStudent without random noise. It provided a gain of 3% in the WCSR score, while the ACQA increased by 23% when compared to the baseline.

6.3 Comparison of Confidence-Boosting Techniques

This section will provide the results of the second experiment, which consisted of comparing different confidence-boosting techniques against three different baselines: one is the same baseline as in the first experiment (using a smaller but strongly labeled training dataset), a second one done with the purpose of measuring the degree of impact of the generated dataset, and a third baseline using the best technique from the previous experiment (i.e., Noisy Student with random noise). The first and second baselines follow the same single iteration training scheme, one using the Isophonics dataset exclusively and the second using both the Isophonics and Generated Pseudolabel dataset.

The last baseline will follow the iterative Noisy Student training scheme presented in previous chapters and use the Isophonics dataset to train the teacher model on the first iteration, and the DALI dataset audios as the large unlabeled dataset. It is important to notice that a second baseline using both the Isophonics and the Generated dataset as input on a single training iteration was necessary, as the Generated dataset will be used in the experiment with the confidence-boosting techniques, so evaluating its impact without these techniques is necessary to have a fair comparison. In the case of the Noisy Student baseline, it is not included, as its corresponding audios are already being used in the iterative training process.

6.3.1 Overall Results

Table 6.2 presents a summary of the results obtained on the best iterations of each different setup. First, it is important to notice that the baselines without Noisy Student are not trained in an iterative process and therefore have only a single entry for each metric. Also, an additional iteration was trained for the NoisyStudent baseline to standardize the experiment, as improvements for the new techniques

tested were still available on the third iteration.

	WCSR				ACQA			
Iteration	1	2	3	4	1	2	3	4
Labeled Baseline	53.5				19.6			
Labeled and Generated Baseline	40.7				13.6			
Noisy Student Baseline	54.8	53.8	52.6	53.6	24.9	26.3	26.3	25.1
Label Equality	55.1	55.5	55.6	54.5	25.1	26.5	26.6	27.7
CFS	55.2	55.6	53.8	52.9	24.4	27.7	24.1	23.4
TIV Similarity	54.5	55.4	54.2	54	23.1	25.6	24.6	23.3

Table 6.2 – Weighted Chord Symbol Recall (WCSR) and Average Chord Quality Accuracy (ACQA) for each experiment and the baselines on the RWC dataset on the second experiment.

Initially, the first thing to notice is that the "Labeled and Generated Baseline", which used the two sets of labels during training, provided much worse results than the rest. In particular, the results were even worse than the "Labeled Baseline", which uses a much smaller training set but with strong labels. Since adding the large, weakly labeled dataset directly to the training set reduces the accuracy, we believe that several of the weak labels are not correct, which is expected due to the nature of the dataset generation. Because of these results, this baseline will not be considered in some of the more detailed analyses presented below to simplify the discussion.

Now, considering the other setups used for this experiment, it is possible to see that the usage of the weak labels improved both the WCSR and ACQA metrics when explored for self-learning. However, it seems that for most of the techniques applied, this improvement started to decay after a few iterations, especially in the WCSR. The best scores for each metric vary along the techniques and iterations, with the best WCSR scores being found in the Label Equality and CFS scenarios in iterations 3 and 2, respectively. Meanwhile, the best ACQA scores are found in these same setups, in iterations 4 and 2, respectively. As the CFS technique provided in a single iteration the best results for each metric, it can be considered the best performing result out of the experiment.

Looking at Table 6.2, the best result provided an increase of 1.5% over the Noisy Student baseline and 3.9% over the Labeled baseline in the WCSR score. Meanwhile, the ACQA score was improved by 5.3% compared to the Noisy Student baseline and by more than 41% when compared to the Labeled Baseline. The Label Equality boosting scheme provided these same improvements, although they

appeared in different iterations, and the TIV Similarity provided improvements over the baselines, but not as significant as the ones in the other two setups.

Similar to the results presented from the first experiment, and in comparison with (CHO, 2014), this new experiment presented in this section provided not only improvements over the baseline in the ACQA score, but it also provided improvements on the WCSR at the same time. Hence, the proposed schemes do not compromise one metric to boost the other, as opposed to (CHO, 2014).

6.3.2 Results by Chord Type

One of the crucial aspects that the techniques presented here aim to achieve is to improve the performance of the classifier in an imbalanced training and evaluation scenario. Most of the metrics presented in the previous sections already reflect this objective. However, it is important to dive deeper into this analysis and show not only the aggregated metrics (such as the WCSR or ACQA), but also the individual performance per chord type. This section will break down the analysis into a more granular level by comparing a per-chord performance.

This analysis starts from Table 6.3, in which the accuracy of some of the techniques presented in previous sections are shown for certain groups of chord types (i.e., the $WCSR_C$). The choice for these chord types was made to provide a varied range of types, regarding their classification difficulty, from the easier ones (such as major and minor) up to some very difficult ones (such as hdim7). Only chords which the selected classifier could support were included here.

	maj	min	7	min7	maj7	dim	hdim7
% of train dataset	63	16.1	6.9	2.6	1	0.4	0.2
% of test dataset	45	15	7.2	13.8	7.4	0.8	0.4
Labeled Baseline	78.1	73	37	15.9	3	11.8	2
Labeled and Generated Baseline	61.1	66.2	12.6	1.9	0	0	0
Noisy Student Baseline	67.3	55.7	56.6	48.6	19.3	28.9	41.3
Label Equality	70.7	51	50.5	56	24.2	28.7	39.7
CFS	69.1	51	52.9	58	27.4	36.4	31.7
TIV Similarity	73	65	46.4	41	13.9	29.7	38.9

Table 6.3 – Accuracy for each of the experiments divided by chord class on RWC dataset for the second experiment

Following the naming conventions of the second experiment, we can see three baselines and the three techniques tested in that experiment. Immediately, it is

possible to observe that, again, the "Labeled and Generated Baseline" provided a degradation in the performance across all chord types when compared to the Labeled baseline, so no further analysis for it will be made. Regarding the Noisy Student baseline and the other improved techniques based on the Noisy Student, it is possible to see a major improvement for rarer chord types (i.e., the ones with less than 10% of the test dataset). However, these same techniques provided worse results for the major and minor chords, although not enough to impact the WCSR score, as seen in the previous sections.

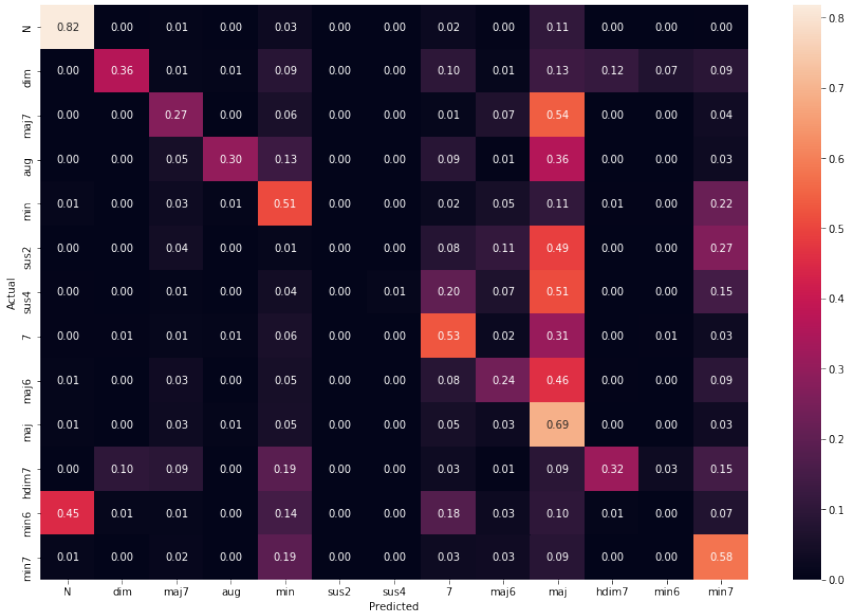
Now, looking in more detail at the techniques proposed in the second experiment, it is possible to see how the one that performed the best both in the WCSR and ACQA scores, the CFS technique, achieved the best results in three out of the five rare chords presented here. On the other hand, it provided one of the worst scores for the remaining chords when compared to the other techniques. For instance, by looking at the Label Equality results, which also obtained very good WCSR and ACQA scores, we can see how the performance is a little bit more balanced across the presented chords and could be a good choice for this purpose.

Meanwhile, the TIV Similarity, which did not provide any improvements over the Noisy Student Baseline, could be considered the worst one at first glance. However, it is interesting to notice that although its results were not the best, it provided an improvement for rare chords over the Labeled Baseline, with the least amount of reduction in the score for the common classes when compared to the other Noisy Student techniques. This means it could also be considered a more balanced choice over CFS if there is a need to keep the common chord scores a bit more elevated.

A second analysis for the individual chord performance can be seen in Figure 6.2, which presents the confusion matrices for the Labeled Baseline and CFS techniques, respectively. Again, some of the possible chords were omitted to make the plot clearer. It is interesting to notice that the confusion matrix of the CFS technique presents a much more expressive diagonal component over the Labeled Baseline. This is a more visual way of seeing how these techniques act upon the classifiers. In the Labeled Baseline, we can see that the predictions are gathered around the major and minor chords, which supports the initial hypothesis that these classifiers are much more biased toward these predominant chord types. On the other hand, the CFS matrix shows a much more distributed set of predictions,

which are much more well distributed along all the available possibilities despite being still gathered around these common chord types.

(a) Labeled Baseline confusion matrix



(b) CFS confusion matrix

Figure 6.2 – Confusion matrices for the baseline and best setup from the second experiment

6.3.3 Performance based on the Presence of Singing-Voice

Finally, another aspect considered important to analyze is the difference in performance for excerpts with and without a singing voice. The reason why this

could be important is based on the way the techniques presented in the second experiment are organized. As the dataset generation technique depends upon the existence of lyrics for matching the chord files and the word timestamps, it is possible that the proposed technique would provide improvements only for these excerpts or even reduce the performance in purely instrumental excerpts.

As the techniques were evaluated on the RWC dataset, it was possible to use an extension by Mauch et al. (2011), which provides the voice and instrumental activity annotations for this same dataset. With this information, it was possible to calculate both the WCSR and ACQA scores exclusively for excerpts with and without lyrics, and allow the comparison between them to be made. The results for the Labeled Baseline, Noisy Student Baseline, and the best-performing iteration of the CFS technique can be seen in Table 6.4. Immediately, it is possible to notice that CFS provided improvements over both baselines in all scenarios considering score types and excerpt types. Another interesting point to note is that the excerpts with singing voice performed slightly worse in general for the presented Baselines, but were the ones with the largest improvements in the CFS scenario. This behavior is actually expected, since the weakly generated dataset contains only audio excerpts with a singing voice. However, it is also able to improve the metrics in Instrumental only sections, which is an interesting characteristic.

		Overall		Singing Voice		Instrumental	
		WCSR	ACQA	WCSR	ACQA	WCSR	ACQA
Labeled Baseline	Scores	53.5	19.6	52.6	19	54.5	19.7
Noisy Student Baseline	Scores	53.8	26.3	53.4	26.1	54.3	26.4
	Change	+0.56%	+34.18%	+1.52%	+37.37%	-0.37%	+34.01%
CFS (Best Iteration)	Scores	55.6	27.3	55.4	27.5	56	26.6
	Change	+3.93%	+39.29%	+5.35%	+44.74%	+2.75%	+35.03%

Table 6.4 – Performance of different techniques when split by instrumental only and singing voice excerpts

7 CONCLUSION

This work presented self-learning strategies in the context of ACR, in particular focusing on the recognition of rare classes. The first approach consisted of using a baseline classifier trained using manually annotated data (strong labels), and then training a student model using a large unlabeled dataset. The second approach extends the first one by automatically generating weak labels for the large dataset using information from online music communities, and then combining these weak labels with the predictions of the teacher to improve ACR.

In the first experiment using unlabeled data only, two different techniques were applied. The first, known as Focal Loss, attempted to improve the performance of complex, hard-to-learn chords by using a modified loss function, without the need for additional data. The second technique explored was the translation of an iterative image training procedure, known as Noisy Student, to the audio domain. This technique was based on the usage of a small labeled dataset, along with a large unlabeled one, and through an iterative self-learning process, aimed to improve and balance the classification results for the proposed setup.

During the second experiment, a modified version of the above-mentioned extension of the Noisy Student algorithm was applied by using not only unlabeled data, but also a set of weak-labels for the same unlabeled dataset. This resulted in a second extension to the Noisy Student technique, in which the weak-labels were used as a way to better select the samples during the self-learning iterative process, by boosting the prediction confidence. Three different musical-aware techniques for the boosting process were tested: Label Equality, Chroma Feature Similarity, and TIV Similarity.

As an additional contribution of this thesis, we presented a technique for automatically obtaining the weakly-labeled dataset. It was based on the usage of song lyrics, on- and off-set information along with guitar community chord labels. As the lyrics could be considered easier to label, because no expert knowledge is required for it, this work proposed using an on- and off-set lyrics dataset in order to evaluate the technique's performance. By merging the time information from this lyrics dataset with the chord labels from the guitar chord community, we made a weakly labeled estimation for the chord on and off-set.

The results presented in this thesis show that it was not only possible to

improve the rare and complex chord performance on its own, but it was also possible to improve the overall classification performance obtained by evaluating all chords together. These results provide a different outcome from previous work that aimed at improving the performance in the same way (DENG; KWOK, 2017), in which the rare chord classification metrics were improved at the cost of the overall results.

Based on the evaluated approaches, it is possible to conclude that the experiment that used both the Noisy Student technique and the weakly labeled dataset using the CFS boosting technique achieved the best results regarding both rare chord and overall performance. Besides, the improvement occurred in excerpts with and without lyrics, which was a concern, considering that the confidence-boosting technique was only usable for excerpts with lyrics. The evaluation of the results provided a better confusion matrix, as it had a much more distinctive diagonal component and only a slight reduction in performance for some of the more common classes.

Regarding future work, there are some different aspects that could be further explored related to the techniques proposed and evaluated here. In the Noisy Student extension, it was necessary to account for the temporal relationship of the audio, which was not present in the original technique, focused on image classification. One possibility for extension would be to further evaluate different techniques for accounting for the sequential aspect of this domain, with different minimum excerpt sizes, or varying the excerpt size based on chord type, to further explore their musical relationships. Also in this direction, it would be possible to evaluate different excerpt selection techniques in order to better balance the self-labeled dataset during the iterative training or to use different techniques to add noise and evaluate their impacts on the performance.

There are also other avenues that could be explored regarding the dataset generation process. One initial idea would be to evaluate better selection techniques for the weak labels, such as combining different chord community annotations, or files from different communities, in order to vote for the label most likely to be correct. In this same line, videos containing song covers or guitar tutorials could also be explored, as they usually provide some image reference for the chord being played. Another aspect to be explored in this same problem would be to properly evaluate the lyric-chord matching process done in order to estimate the weak labels. This would allow different dataset generation techniques to be evaluated and possibly

discover better ways of achieving this dataset.

Finally, regarding the second extension to the Noisy Student technique involving confidence boosting, it is also left for future work to evaluate different confidence-boosting techniques and their combinations, besides the ones already tested. Also, the overall experiment could also be further extended by combining the techniques with different classifiers and dataset combinations to further assess its performance under different conditions.

REFERENCES

BEERY, S. et al. Synthetic examples improve generalization for rare classes. In: The IEEE Winter Conference on Applications of Computer Vision. [S.l.: s.n.], 2020. p. 863–873.

BEKKER, A. J.; GOLDBERGER, J. Training deep neural-networks based on unreliable labels. In: IEEE. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.], 2016. p. 2682–2686.

BERENZWEIG, A. et al. A large-scale evaluation of acoustic and subjective music-similarity measures. Computer Music Journal, JSTOR, p. 63–76, 2004.

BERNARDES, G. et al. A multi-level tonal interval space for modelling pitch relatedness and musical consonance. Journal of New Music Research, v. 45, p. 1–14, 05 2016.

BOULANGER-LEWANDOWSKI, N.; BENGIO, Y.; VINCENT, P. Audio chord recognition with recurrent neural networks. In: CITESEER. ISMIR. [S.l.], 2013. p. 335–340.

BROWN, J. C. Calculation of a constant q spectral transform. The Journal of the Acoustical Society of America, Acoustical Society of America, v. 89, n. 1, p. 425–434, 1991.

BUDA, M.; MAKI, A.; MAZUROWSKI, M. A. A systematic study of the class imbalance problem in convolutional neural networks. Neural Networks, Elsevier, v. 106, p. 249–259, 2018.

BURGOYNE, J. A.; WILD, J.; FUJINAGA, I. An expert ground truth set for audio chord recognition and music analysis. In: ISMIR. [S.l.: s.n.], 2011. v. 11, p. 633–638.

CHEN, T.-P.; SU, L. Attend to chords: Improving harmonic analysis of symbolic music using transformer-based models. Transactions of the International Society for Music Information Retrieval, Ubiquity Press, v. 4, n. 1, 2021.

CHEW, E. The spiral array: An algorithm for determining key boundaries. Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), v. 2445, 07 2002.

CHO, T. Improved techniques for automatic chord recognition from music audio signals. Thesis (PhD) — New York University, 2014.

CUBUK, E. D. et al. Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. [S.l.: s.n.], 2020. p. 702–703.

DENG, J.-q.; KWOK, Y.-K. Large vocabulary automatic chord estimation with an even chance training scheme. In: ISMIR. [S.l.: s.n.], 2017. p. 531–536.

EMIYA, V. et al. Maps-a piano database for multipitch estimation and automatic transcription of music. 2010.

FUJISHIMA, T. Real-time chord recognition of musical sound: A system using common lisp music. Proc. ICMC, Oct. 1999, p. 464–467, 1999.

GAO, X.; GUPTA, C.; LI, H. Automatic lyrics transcription of polyphonic music with lyrics-chord multi-task learning. IEEE/ACM Transactions on Audio, Speech, and Language Processing, IEEE, v. 30, p. 2280–2294, 2022.

GOTO, M. et al. Rwc music database: Popular, classical and jazz music databases. In: ISMIR. [S.l.: s.n.], 2002. v. 2, p. 287–288.

HAAS, W. de et al. Comparing approaches to the similarity of musical chord sequences. In: . [S.l.: s.n.], 2010. v. 6684, p. 242–258. ISBN 978-3-642-23125-4.

HAN, B. et al. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In: Advances in neural information processing systems. [S.l.: s.n.], 2018. p. 8527–8537.

HARTE, C. Towards automatic extraction of harmony information from music signals. Thesis (PhD), 2010.

HARTE, C. et al. Symbolic representation of musical chords: A proposed syntax for text annotations. In: ISMIR. [S.l.: s.n.], 2005. v. 5, p. 66–71.

HINTON, G.; VINYALS, O.; DEAN, J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

HUANG, G. et al. Deep networks with stochastic depth. In: SPRINGER. European conference on computer vision. [S.l.], 2016. p. 646–661.

HUMPHREY, E. J.; BELLO, J. P. Rethinking automatic chord recognition with convolutional neural networks. In: IEEE. 2012 11th International Conference on Machine Learning and Applications. [S.l.], 2012. v. 2, p. 357–362.

JIANG, J. et al. Large-vocabulary chord transcription via chord structure decomposition. In: ISMIR. [S.l.: s.n.], 2019. p. 644–651.

KEHLING, C. et al. Automatic tablature transcription of electric guitar recordings by estimation of score-and instrument-related parameters. In: DAFx. [S.l.: s.n.], 2014. p. 219–226.

KORZENIOWSKI, F.; WIDMER, G. A fully convolutional deep auditory model for musical chord recognition. In: IEEE. 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP). [S.l.], 2016. p. 1–6.

LAITZ, M. R. C. S. G. The Complete Musician: An Integrated Approach to Theory, Analysis, and Listening. [S.l.]: Oxford University Press, 2022. ISBN 9780190924508.

LEE, K. A system for automatic chord transcription from audio using genre-specific hidden markov models. In: SPRINGER. International Workshop on Adaptive Multimedia Retrieval. [S.l.], 2007. p. 134–146.

LERDAHL, F. Tonal pitch space. Music Perception: An Interdisciplinary Journal, University of California Press, v. 5, n. 3, p. 315–349, 1988. ISSN 07307829, 15338312. Available from Internet: <http://www.jstor.org/stable/40285402>.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet physics doklady. [S.l.: s.n.], 1966. v. 10, n. 8, p. 707–710.

LI, X. et al. Learning to self-train for semi-supervised few-shot classification. In: Advances in Neural Information Processing Systems. [S.l.: s.n.], 2019. p. 10276–10286.

LI, Y. et al. Learning from noisy labels with distillation. In: The IEEE International Conference on Computer Vision (ICCV). [S.l.: s.n.], 2017.

LIN, T.-Y. et al. Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2017. p. 2980–2988.

MARQUES, L. A chord distance metric based on the tonal pitch space and a key-finding method for chord annotation sequences. In: Anais do XVII Simpósio Brasileiro de Computação Musical. Porto Alegre, RS, Brasil: SBC, 2019. p. 136–140. ISSN 0000-0000. Available from Internet: <https://sol.sbc.org.br/index.php/sbcm/article/view/10435>.

MAUCH, M.; FUJIHARA, H.; GOTO, M. Integrating additional chord information into hmm-based lyrics-to-audio alignment. IEEE Transactions on Audio, Speech, and Language Processing, IEEE, v. 20, n. 1, p. 200–210, 2011.

MAUCH, M. et al. Timbre and melody features for the recognition of vocal activity and instrumental solos in polyphonic music. In: ISMIR. [S.l.: s.n.], 2011. p. 233–238.

MCFEE, B.; BELLO, J. P. Structured training for large-vocabulary chord recognition. In: ISMIR. [S.l.: s.n.], 2017. p. 188–194.

MCFEE, B.; HUMPHREY, E. J.; BELLO, J. P. A software framework for musical data augmentation. In: ISMIR. [S.l.: s.n.], 2015. v. 2015, p. 248–254.

MESEGUER-BROCAL, G.; COHEN-HADRIA, A.; PEETERS, G. DALI: A Large Dataset of Synchronized Audio, Lyrics and notes, Automatically Created using Teacher-student Machine Learning Paradigm. In: Proceedings of the 19th International Society for Music Information Retrieval Conference. Paris, France: ISMIR, 2018. p. 431–437. Available from Internet: <https://doi.org/10.5281/zenodo.1492443>.

MÜLLER, M. Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications. [S.l.]: Springer International Publishing, 2016. ISBN 9783319357652.

ODEKERKEN, D.; KOOPS, H. V.; VOLK, A. DECIBEL: Improving Audio Chord Estimation for Popular Music by Alignment and Integration of Crowd-Sourced Symbolic Representations. 2020.

OUDRE, L.; GRENIER, Y.; FÉVOTTE, C. Template-based chord recognition: Influence of the chord types. In: ISMIR. [S.l.: s.n.], 2009. p. 153–158.

PARK, J. et al. A Bi-Directional Transformer for Musical Chord Recognition. In: Proceedings of the 20th International Society for Music Information Retrieval Conference. Delft, The Netherlands: ISMIR, 2019. p. 620–627. Available from Internet: <https://doi.org/10.5281/zenodo.3527886>.

PAUWELS, J. et al. 20 years of automatic chord recognition from audio. In: International Society for Music Information Retrieval (ISMIR). Delft, Netherlands: [s.n.], 2019. Available from Internet: <http://hdl.handle.net/10230/42773>.

ROWE, L. O.; TZANETAKIS, G. Curriculum Learning for Imbalanced Classification in Large Vocabulary Automatic Chord Recognition. In: Proceedings of the 22nd International Society for Music Information Retrieval Conference. Online: ISMIR, 2021. p. 586–593. Available from Internet: <https://doi.org/10.5281/zenodo.5624463>.

SALAMON, J.; JACOBY, C.; BELLO, J. P. A dataset and taxonomy for urban sound research. In: 22nd ACM International Conference on Multimedia (ACM-MM'14). Orlando, FL, USA: [s.n.], 2014. p. 1041–1044.

SHEH, A.; ELLIS, D. P. Chord segmentation and recognition using em-trained hidden markov models. In: Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR). [S.l.: s.n.], 2003. p. 183–189.

SRIVASTAVA, N. et al. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014.

Wu, Y.; Li, W. Automatic audio chord recognition with midi-trained deep feature and blstm-crf sequence decoding model. IEEE/ACM Transactions on Audio, Speech, and Language Processing, v. 27, n. 2, p. 355–366, 2019.

Wu, Y.; Li, W. Automatic audio chord recognition with midi-trained deep feature and blstm-crf sequence decoding model. IEEE/ACM Transactions on Audio, Speech, and Language Processing, v. 27, n. 2, p. 355–366, 2019.

XI, Q. et al. Guitarset: A dataset for guitar transcription. In: ISMIR. [S.l.: s.n.], 2018. p. 453–460.

XIE, Q. et al. Self-training with noisy student improves imagenet classification. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. [S.l.: s.n.], 2020. p. 10687–10698.

Yadati, K. et al. Detecting socially significant music events using temporally noisy labels. IEEE Transactions on Multimedia, v. 20, n. 9, p. 2526–2540, 2018.

APPENDIX A — RESUMO EXPANDIDO EM PORTUGUÊS

O reconhecimento de acordes musicais é uma tarefa complexa, não somente para algoritmos, mas também para pessoas. O campo de estudo para tarefas relacionadas a musica em geral na ciência da computação se chama Music Information Retrieval (MIR) e é responsável, entre outras coisas, pelo reconhecimento de acordes. Nesse contexto, este trabalho fornecerá sua contribuição, ao explorar técnicas de treinamento, auto-aprendizagem (self-learning) e geração de rótulos fracos (weak labels) para melhorar a performance no reconhecimento de acordes raros.

Estes acordes raros são em geral sub-representados nos conjuntos de dados para treinamento de classificadores. Por causa disso, um certo viés acaba sendo gerado, tanto na capacidade de aprendizagem do modelo, quanto na sua performance de validação, sempre beneficiando acordes mais comuns. Diferentes estilos musicais tem diferentes necessidades em termos de tipos de acorde a serem reconhecidos, podendo um acorde considerado raro no contexto dos conjuntos de dados de treinamento, ser extremamente comum em um estilo específico. Por causa disso, as contribuições deste trabalho visam melhorar de maneira global essa performance, além de trazer resultados mais balanceados, podendo classificadores como esse serem utilizados em aplicações comerciais e educacionais de transcrição de acordes, onde uma performance cada vez melhor de reconhecimento traz grandes benefícios.

A tarefa de reconhecimento de acordes tem sua origem com algoritmos muito dependentes de um conhecimento pré-programado neles, como em Fujishima (1999), Sheh and Ellis (2003), Lee (2007). Porém, ao longo do tempo, houve uma transição para técnicas mais voltadas para uso em massa de dados e deep learning (PAUWELS et al., 2019). Nesse contexto diversas técnicas foram exploradas, como Deep Neural Networks (BOULANGER-LEWANDOWSKI; BENGIO; VINCENT, 2013), arquitetura de encoder-decoder e Convolutional Neural Network (MCFEE; BELLO, 2017). Além disso, algumas técnicas ainda mais recentes buscam explorar a forte relação temporal existente no contexto de áudio musical. Nesse cenário podemos observar técnicas como Long Short-Term Memory foram usadas (DENG; KWOK, 2017; Wu; Li, 2019a) e o uso de arquitetura do tipo Transformer (PARK et al., 2019; CHEN; SU, 2021).

Mais específicamente para o reconhecimento de acordes raros, temos também muitos trabalhos relevantes relacionados, focando principalmente na utilização

de arquiteturas voltadas para trazer um melhor resultado em problemas desbalanceados, como em (Wu; Li, 2019a; MCFEE; BELLO, 2017; JIANG et al., 2019). Além disso, também é importante mencionar duas técnicas que aplicaram o uso de rótulos provenientes de comunidades online para anotação de acordes em (MAUCH; FUJIHARA; GOTO, 2011; ODEKERKEN; KOOPS; VOLK, 2020), pois também será algo relevante no contexto desse trabalho.

O trabalho apresentado aqui atuou em duas frentes, ambas focadas no reconhecimento de acordes raros. Primeiramente, um algoritmo para a geração de rótulos fracos para acordes é apresentado. Em segundo lugar, diferentes técnicas de treinamento para classificadores são testadas, voltadas para a melhoria de performance nesses acordes considerados raros.

A técnica de geração de rótulos é desenvolvida com base no uso de rótulos de acordes obtidos em comunidades online, onde diferentes usuários podem fornecer a relação entre acordes e o texto da letra de uma música em formato de texto. Com base nessa relação, adicionado de um componente extra, que é a associação entre cada palavra de uma música e o momento preciso de seu início e fim, é possível extrapolar o momento aproximado em que cada acorde deveria iniciar e terminar. Apesar dessa informação não estar disponível imediatamente, espera-se que rotular esses momentos precisos associados a letra da música, por ser uma tarefa sem necessidade de especialista, seria algo menos custoso ao se gerar um novo conjunto de dados. Para o contexto desse trabalho, essa informação foi obtida através de um conjunto de dados pré-existente, chamado DALI (MESEGUER-BROCAL; COHEN-HADRIA; PEETERS, 2018). Também é importante mencionar que a técnica proposta funciona apenas em trechos onde a letra da música está presente, pois há essa dependência.

Para as técnicas de treinamento de classificador foram exploradas algumas diferentes alternativas. Uma primeira técnica, conhecida como Focal Loss (LIN et al., 2017), visa trazer uma melhor performance em exemplos mais difíceis de treinamento, ao modificar a componente da loss, adicionando um componente exponencial. Esse componente diminui o impacto da componente da loss de exemplos que possuem uma maior confiança na predição do modelo, aumentando o peso de exemplos com menor confiança, como seriam o caso de acordes mais raros.

As outras duas técnicas testadas, focaram no uso de uma técnica de treinamento iterativo com auto-aprendizagem, conhecido como Noisy Student (XIE et al.,

2020). Essa técnica utiliza dados não-rotulados para tentar aumentar a gama de exemplos de um dataset de treinamento, trazendo ao mesmo tempo um maior balanceamento e melhoria de performance. Originalmente essa técnica foi desenvolvida para o contexto de reconhecimento de imagem, portanto, também foi desenvolvida nesse trabalho uma adaptação dessa técnica para melhor se adaptar ao contexto de áudio e música, principalmente passando a levar em conta o contexto temporal intrínseco deste domínio.

Utilizando essa mesma técnica, uma segunda extensão foi adicionada, na qual os dados gerados utilizando acordes de comunidades online foram incluídos no processo iterativo de treinamento da Noisy Student. Nessa adaptação, diferentes técnicas de aumento de confiança de predição foram utilizadas no processo de auto-aprendizagem, para melhorar o processo iterativo de seleção existente. Com isso, a expectativa é obter uma melhor seleção de exemplos ao longo do processo, o que traria um melhor resultado também na performance final do classificador. Três diferentes técnicas para melhoria de confiança foram testadas, com base na igualdade de label ou diferentes graus de similaridade calculados com base nos conceitos musicais por trás dos acordes.

Para o resultado dos experimentos e sua avaliação, além da métrica mais comum aplicada no contexto de reconhecimento de acordes, que fornece uma performance global, também foi feita uma avaliação com uma métrica alternativa. Essa outra métrica produz um resultado levando em consideração um peso igual da performance de cada acorde, independente do percentual que ele representa no conjunto de dados. Todas as técnicas aplicadas nesse trabalho produziram ganhos significativos de performance nas métricas balanceadas. Além disso, as técnicas baseadas no Noisy Student geraram também uma melhoria na métrica de performance geral, em especial as técnicas desenvolvidas utilizando o conjunto de dados gerado e aumento de confiança no processo iterativo. Este último cenário trouxe os melhores resultados entre todos os experimentos feitos, trazendo também uma comprovação da eficácia do processo de geração de rótulos fracos.

Para trabalhos futuros restam algumas avenidas a serem exploradas. Para o processo de seleção do Noisy Student, novas técnicas de seleção iterativa poderiam ser testadas no contexto de áudio, além de novas técnicas de aumento de confiança. Além disso, para a geração do dataset, poderíamos explorar outras alternativas, como por exemplo combinar diferentes fontes de dados, ou utilizar videos educativos

com exemplos visuais para extrair os rótulos. Nesse mesmo contexto, poderíamos também explorar uma maneira mais concreta de avaliar o resultado da geração dos rótulos fracos, que apenas foi validado indiretamente ao ser utilizado pela técnica de treinamento.