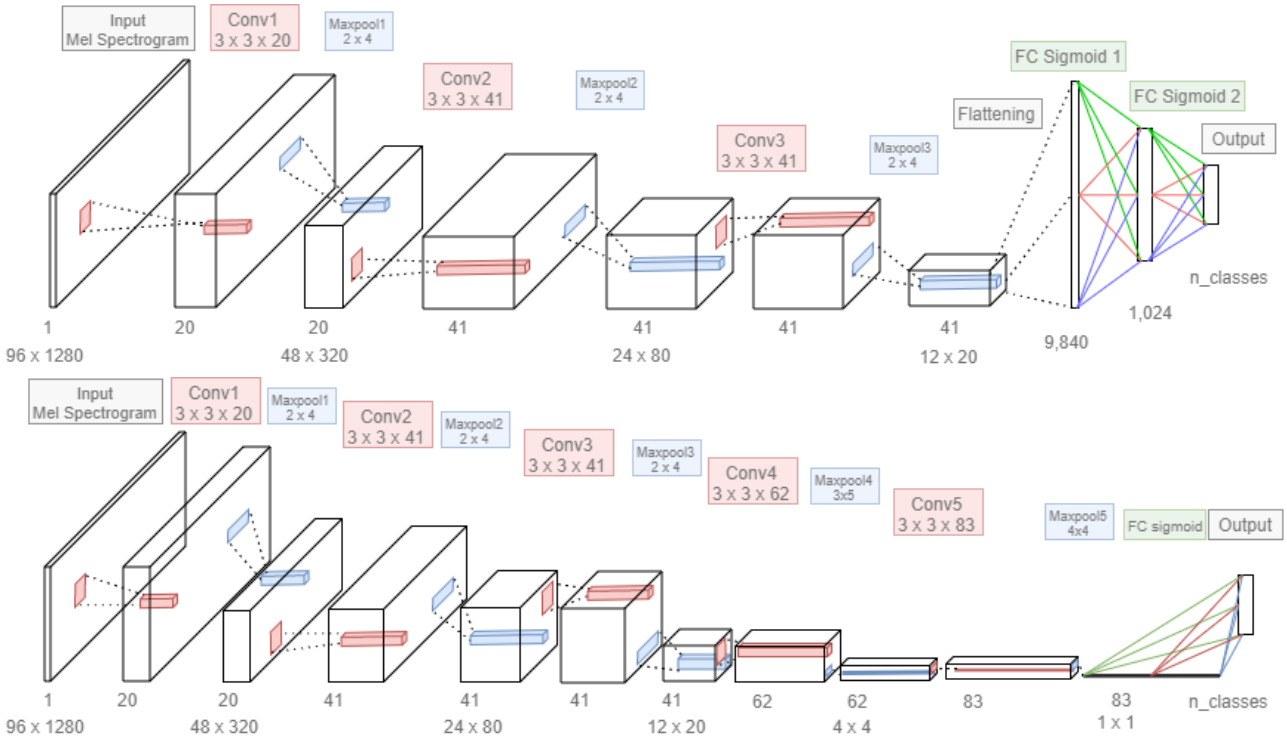


Electronic Music Genre Recognition

Electronic Subgenre Recognition

Using Convolutional Neural Networks

Joram Wessels

Electronic Music Genre Recognition

Electronic Subgenre Recognition Using Convolutional Neural Networks

Joram Wessels 10631542

Thesis Project

Bachelor Artificial Intelligence

College of Science
University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervised by

dr. ir. Rein van den Boomgaard
drs. Taco Walstra

Informatics Institute
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

Special thanks to Bas Visser
for his help during technical difficulties

June 25th, 2017

Abstract

The presented work proposes a new method of classifying music genres. The field of music genre recognition (MGR) aims to develop a system that can predict the genre of any song. However, the past 20 years of research on this topic have taken certain variables for granted, such as the size of the datasets and the genres used for classification. Calling these variables into question could discover valuable directions for the entire field. Since music genres are structured like a taxonomy, one could train a model on the subgenres of the genres used for testing by abstracting the predictions back to their main genre. This allows the model to learn about the specific ways a genre is manifested. Electronic music is a suitable testing ground for such an inquiry, since it is saturated with taxonomical subdivisions, the resulting classifier is relevant in contemporary society, and yet insufficient research has been conducted on this genre. No prior research was found using a neural network to classify electronic subgenres. Two convolutional neural networks (CNN) models and two different datasets have been applied to test the effect of subgenre targeting. It achieved moderately positive results, only being beneficial for larger models and larger datasets. The best accuracy was 92.4% on a binary classification task using no subgenre targeting. Future work could evaluate the effects of subgenre targeting on larger models and datasets while controlling for the amount of parameters.

Contents

1 Introduction

Music genre recognition (MGR) is an application of machine learning within field of Music Information Retrieval (MIR). The ultimate goal is to create a system that can perfectly identify the genre of any song. There is a high demand for such a system among music retailers like Spotify or Amazon, which is why these companies sponsor the MIR contests that drive the research in this field [19]. Traditional approaches to MGR experimented with selecting the right features that best classified a few of the most general genres (rock, classical, blues, etc.) [39], predominantly using support vector machines (SVM), to learn the associations. But since the breakthrough of convolutional neural networks [24] (CNN) in 2012, MGR has experimented a lot with neural network approaches, CNNs being the most popular type [10, 23, 27, 28], followed by Recurrent Neural Networks (RNN) [11, 18, 36].Another development of the last decade that directs the field of MGR is the increased popularization of electronic music (sometimes referred to as 'EDM' or 'dance', but the synonymy of these terms is debated [34]). As electronic music has become more relevant in society, recognition of its subgenres has become more relevant to the aforementioned music retailers as well. Additionally, companies specifically affiliated with electronic music, like Native Instruments [19], further upped the ante for for an electronic music classifier. However, despite this rise in popularity, little research has been conducted on the recognition of electronic music. Some datasets used in previous work included 'Electronic' as a target variable, yet such an inclusion is a misrepresentation of the genre. Electronic music is simply too diverse to be treated like a composite. A handful of studies refrained from doing so, focusing exclusively on electronic subgenres [6, 22, 29], but these works exhibited problems of their own. The datasets were substantially smaller than the already exceptionally small datasets used in MGR and paid little attention to the diversity of artists. Furthermore, none of these studies have applied a neural network.

Therefore, there is a discrepancy between the general research on MGR and the specific research on electronic subgenres. The goal of the presented research is to unite the results of these two novel paradigms into a single system that can recognize electronic subgenres using a state of the art neural network algorithm. This will determine to what extent convolutional neural networks can improve the accuracy of electronic subgenre classification when targeting subgenres individually rather than as a collective. The resulting specialized MGR classifier may at one point serve as a module in a potential future multi-agent MGR system, or could alternatively be used independently in an application that facilitates DJing.

2 Related Work

Most centralization and coordination of MIR research is due to the efforts of ISMIR, the International Society of Music Information Retrieval. Its annual conference hosts the MIREX contest in which algorithms submitted by researchers face each other off in a number of categories. The three categories relevant to the presented work are 'Mixed Popular Genre Classification', 'Latin Genre Classification' and 'K-Pop Genre Classification'. This decade, all three have effectively been dominated by the research of Ming-Ju Wu and Jyh-Shing Roger Yang [44], winning at least one of the categories each year since 2012. Their approach took into account the spectrogram features themselves, as well as their temporal variations. It was Aggelos Pikrakis who in 2013 submitted the first neural network approach and subsequently won the Latin Genre Classification challenge. Following his success, many more papers were published using novel neural network attempts, including a probabilistic neural network (PNN) [29], a multilayer perceptron (MLP) [15], and a radial basis function network (RBF) [1], but none improved on Pikrakis' achievement, let alone on the annually improving algorithm by Wu.

The 2016 paper by Thomas Lidy and Alexander Schindler [28] was the first MIREX submission using a CNN approach and managed to win in two categories. It should be noted, however, that Lidy had only one competitor in both the K-Pop and Latin Genre categories [14], and that he still lost to Wu in the Mixed Genre Classification. The first use of CNNs in MGR research was conducted by Tom Li in 2010 [27], but he achieved an accuracy less than 30% on the ten classes of the GTZAN dataset [41]. Qiuqiang Kong was more successful in 2014, scoring a 72.4% accuracy on that same dataset, whereas Lidy's 2016 victory was based on a mere 69.54% accuracy, and achieved a similar accuracy of 73.14% on the MIREX mixed genre dataset of 10 classes. However, the comparison of these results might be unjustified since they were not derived from the same dataset.

Another notable neural network used in MGR research is the RNN. Although its purposes had already been researched for other MIR tasks since 2014 [36], the RNN was only applied to MGR last year. Jeremy Irvin used a Long Short-Term Memory (LSTM) RNN without attention on the GTZAN dataset and achieved a 79% accuracy [18]. Jia Dai followed with another LSTM implementation and scored 89.71% accuracy on the ISMIR dataset [11]. Two other studies managed to combine a CNN and an RNN into a so called CNN-RNN hybrid. Benedikt Vogler, from the university of Weimar, proposed the concatenation of a CNN with an LSTM and tested his implementation on a custom unreproducible dataset of 3 genres [42], resulting in an 83% accuracy. Volger did not provide any specifics on how exactly the two networks were concatenated, and instead appeared to focus on explaining only the basics of neural networks. The second study is that of Keunwoo Choi et al. of the Queen Mary University [8]. Choi engineered a 'CRNN' and compared it with three normal CNN implementations using 50 genres from the Million Song Dataset [5]. He did not report a resulting accuracy for his implementation, but instead provided a graph with AUC-ROC scores ranging from 0.7 to 0.95. He concluded that CRNNs are

comparable to CNNs in accuracy, but trade off computation time for memory.

Whereas MIREX tends to the classification of Latin music and K-pop, and some research even goes as far as to specialize in classifying Malay [16], Cretan or African music [3], little attention has been given to electronic music. Still, the popularity of this genre in today's society should be sufficient incentive to dedicate specialized research to it [2]. Furthermore, its subgenres are mainly defined by its rhythmic structure [32], and such a formal definition of the target is useful when designing the MGR approach. Yet, only four papers have specialized in electronic music genre recognition. In 2007, Priit Kirss implemented a rhythm based classifier as his master's thesis and reportedly achieved a 96.4% accuracy [22], but his custom dataset of 5 genres and only 250 tracks render the results unreliable. Siva et al. trained a PNN in 2014 to classify four electronic subgenres with a 94.16% accuracy, but used an equally small database of only 280 tracks [29]. Austin Chen classified three subgenres for his master's thesis in 2014 and achieved a 80.67% accuracy on a dataset of 30 tracks. Finally, Panteli et al. from the University of Amsterdam used a dataset of 20 tracks and achieved a VAF of 0.41, which they considered a "moderate accuracy" [30].

Due to the use of many different datasets, the results of most studies are incomparable. The datasets used in the MIREX competitions contain 70-100 tracks per genre, but are not disclosed for training purposes. Additionally, the most popular standardized dataset has been subjected to criticism. Bob Sturm evaluated the contents of the GTZAN dataset and found that 7.2% of the excerpts are from the same track (5% being exact duplicates) and 10.8% of them are mislabeled [37]. That being said, many custom datasets are substantially smaller than most machine learning datasets and exhibit little diversity of artists, which directly undermines the reliability of the reported results.

3 Theoretical Background

3.1 Deep Learning

Essentially, any artificial neural network with more than one hidden layer can be considered a deep neural network. The earliest conceptualization of deep learning dates back to the 1960's [20], although no functional implementations were realized until Yann LeCun's application of the backpropagation algorithm in 1989 [25]. Still, 'vanilla' artificial neural networks did not pose a threat to contemporary machine learning methods. It took another two decades of graphics processing unit development before deep learning had a platform to be used on. The turning point of the neural network paradigm was Alex Krizhevsky's deep learning CNN in 2012 called AlexNet [24]. AlexNet proved fundamentally better at image recognition, which incited the current trend of successfully applying deep neural nets to any machine learning problem.

3.1.1 CNN

The first functional Convolutional Neural Network was realized by LeCun in 1998 [26]. Based on the neural pathways in the visual cortex, it has been particularly successful in image recognition due to the scale- and location invariance of decreasing dimensions. A CNN is composed of one or multiple convolutional layers that, as the name suggests, effectively mimic a convolution of the input layer by means of neural connections. A filter is convolved with the input layer to construct the next layer, as is illustrated in 3.1 for a two-dimensional input layer. Each layer of filters, or 'kernels' as they are also known, has its own size and a stride, the amount with which the filter shifts during the convolution. Most models apply zero-padding, a method of adding half of the filter size to the edges of the input frames as zeros, so the pixels at the edges will not be skipped. The composition of the filter can be considered a feature in the traditional sense of machine learning. Once more of these convolutional layers are placed in a series the CNN will exhibit its typical behavior. The filters in the first layer will detect low-level features like edges and blobs and report the presence of these features in the activation map. The filters in the second layer will combine contiguous low-level features into features of a higher level, e.g. a square of 9 nodes in the activation map with detected edges in the outer nodes and nothing

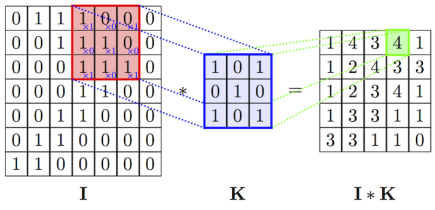


Figure 3.1: A convolution of input layer I with filter K to construct the first hidden layer. (source: CambridgeSpark.com)

in the inner node will decode a circle, and as such this layer will detect slightly more complex features like geometric shapes. Each filter creates an activation map, which means that the output of a convolutional layer with multiple filters is three-dimensional, where the depth represents the channel. The layer usually includes ReLu activation $ReLu(x) = max(0, x)$ but these are sometimes considered to be a separate ReLu layer.

Finally, like all artificial neural networks, the learning occurs during backpropagation. In the case of CNNs, the filters are the parameters to be learned. Each of the filters, which are randomly initialized at the beginning of training, will be specialized to detect a single feature that appears to define the structure of the input.

CNN models are usually enriched with layers other than the convolutional layer, such as pooling layers, dropout layers or fully connected layers. Pooling layers decrease the resolution of the input frames, which lowers the amount of parameters and give the network its scale invariance. There are two pooling options commonly used. Max pooling takes only the highest value from a given window, whereas general pooling takes the mean of the whole window. Dropout can randomly disable some nodes during backpropagation to prevent overfitting, and is sometimes considered to be a property of the learning algorithm rather than a layer. When two nodes are always activated together, they are forced to share the gradient during backprop. But when one of these nodes is disabled by the dropout layer, the active node is no longer overshadowed by the other. Dropout also lowers complexity, since there are less nodes to be trained. A fully connected layer is the traditional kind of neural network layer found in a multilayer perceptron. In CNNs they are usually only found at the very end of the network to convert the output frames of the convolutions to a single output array of the required size. For the last couple of years it has become more common to use softmax activation for this layer

$$\sigma(z)_j = \frac{e_{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

which normalizes the result using the sum of all K exponents and allows them to be interpreted as probabilities or confidence scores.

3.1.2 RNN

The Recurrent Neural Network began with Jürgen Schmidhuber's Neural History Compressor in 1992 [35]. Contrary to other deep learning algorithms, RNNs do not use feed-forward propagation. Whatever the layer outputs is directed back into that same layer as the next input, creating a time loop that can act as a temporary memory. This way they are especially fit for sequences of data and are therefore applied frequently for signal processing tasks. An implementation of just one layer is quite common, but they are often visualized as an unrolled series of time steps like in figure 3.2.b. The horizontal arrows are the cyclic connections to itself, and the horizontal progress is therefore a series of time steps. You can image an RNN of one hidden node as a jump predictor in a CPU; based on the jump in the previous cycle, the CPU predicts another jump. For a standard RNN, the hidden state at time step t is $h_t = \sigma(W^{(hh)}h_{t-1} + W^{(hx)}x_t)$, where W is the matrix of weights, x_t is the input at time step t, and σ is the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. The output $\hat{y}_t$ is the weighted softmax activation of the last hidden state, $\hat{y}_t = softmax(W^{(s)}h_t)$ (the circumflex

indicates y is an estimation rather than a true value). However, RNNs have to cope with what is called the 'vanishing gradient problem', which gets exponentially worse in RNNs. The cyclic connections cause back-propagation to be a lot more computationally intensive, and as a result the gradients become smaller in the long term, while favoring the short term. A popular fix for this problem is called gating, telling each node what to remember and what to forget. The two common gating algorithms are the LSTM and the GRU.

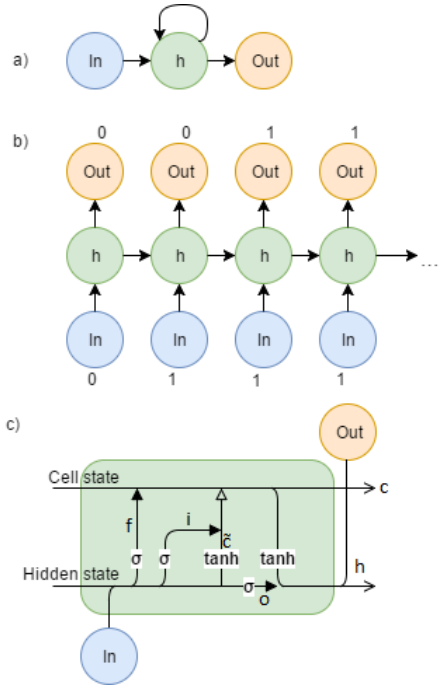


Figure 3.2: a) A simple RNN shown in a traditional way b) The unrolled version of that same RNN predicting program jumps c) The contents of a single LSTM node. Intersections with filled arrows are multiplications, and the intersection with the empty arrow is an addition.

3.1.2.1 LSTM

Proposed by Sepp Hochreiter and Jürgen Schmidhuber in 1997 [17], the Long Short-term Memory is an RNN with slightly different nodes. The process is controlled by the cell state (the upper most horizontal line in figure 3.2.c), which runs like a conveyor belt through all nodes. The cell state can be altered by the input, but the extent to which any individual input can do so is regulated by gates (indicated with a sigma). The gates are element wise multiplications with a sigmoid function, and each node contains three different gates. The 'forget gate' f_t decides what portion of the cell state is maintained given the input and hidden state, $f_t = \sigma(W^{(fx)}x_t + W^{(fh)}h_{t-1})$. The 'input gate' i_t then decides which entries of the input to add to the cell state using a similar formula, $i_t = \sigma(W^{(ix)}x_t + W^{(ih)}h_{t-1})$, which is multiplied by the vector of candidate values $\tilde{c}_t = tanh(W^{(cx)}x_t + W^{(ch)}h_{t-1})$ (the tilde indicates c is an empirical observation). At this point, the cell state $c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$ (where the $\circ$ indicates element wise multiplication). Finally, the 'output gate' $o_t = \sigma(W^{(ox)}x_t + W^{(oh)}h_{t-1})$ decides how much of the cell state c_t is included in the output $h_t = o_t \circ tanh(c_t)$. When the input- and output gates are vectors of only ones and the forget gate is the null vector, the LSTM becomes a regular RNN squashed by the *tanh* function (the hyperbolic tangent, *tanh*, is like a sigmoid function ranging from -1 to 1).

The LSTM has become the standard implementation of RNNs [9] due to its success in tackling the vanishing gradient, and consequent successes in natural language processing, handwriting recognition and speech recognition. A modification of this algorithm, called the seq2seq model, appends a second inversed LSTM to the output nodes in order to use temporal dependencies in the output as well. This is the basis of the new Google Neural Machine Translation system that powers the improved Google Translate [45].

3.1.2.2 GRU

Gated Recurrent Units were proposed by Kyunghyung Cho in 2014 [7]. It could be considered a simplification of the LSTM, in that it combines the three gates into two: the 'reset gate' and the 'update gate'. It also discards the difference between the cell state and the hidden state. The reset gate $r_t = \sigma(W^{(rx)}x_t + W^{(rh)}h_{t-1})$ determines the influence of the previous hidden state on the value candidates $\tilde{h}_t = tanh(W^{(x)}x_t + r_t \circ W^{(h)}h_{t-1})$. The update gate $z_t = \sigma(W^{(zx)}x_t + W^{(zh)}h_{t-1})$ mediates between the current candidates and the previous hidden state, such that the output $h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \tilde{h}_t$. GRU RNNs have been shown to perform similar to LSTMs at a couple of tasks, while requiring less parameters [9]. Since it also deals with the vanishing gradient problem, it might seem obvious that the GRU is better than the LSTM. However, the latter has existed since 1997 and its behaviour has been researched extensively, whereas the former is relatively new and unpredictable. It will require more research on the effect of decreasing the amount of parameters before it can potentially completely replace the LSTM.

3.1.3 Tensorflow

There are three popular libraries that facilitate machine learning applications: Udacity, Theano and Tensorflow. The developers at Google Brain envisioned Tensorflow as a standardized solution for deep learning applications across all devices, which simplifies any changes in hardware during a project. Implementing a neural network has been diminished to a description of a data flow chart. All data is confined with the data structures known as tensors, hence the name of the library. Input parameters are substituted by placeholder objects, and training parameters by variable objects. Implementing a network starts by describing the operations performed on the placeholders. After that, the structure of the network is given by a sequence of these operations as if they are the layers. The amount of nodes in each layer is deduced from dimensions in the variable objects. After these requirements are met, the network comes to life by running it in a session and feeding the input parameters to the placeholder objects. The session object then compiles the high-level description and performs the operations. The back end is handled entirely by Tensorflow, and so running an extensive network on a GPU raises no further complications besides installing the correct drivers and Tensorflow GPU distribution.

3.2 Mel Spectrogram

A sound file such as the mp3 is a series of samples, often in dual channel, that represent a sound wave. In order to make the input compatible with a neural network, these raw data files have to be preprocessed, traditionally to extract features. However, convolutional neural networks depend on images as input variables. Therefore, MIR systems using CNNs convert the sound files to their Mel spectrogram. A spectrogram is a visual representation of the changing power of individual frequencies throughout time. The horizontal coordinate of a pixel determines its place in the audio fragment, the vertical coordinate determines the frequency range it represents, and the

color of the pixel indicates the presence or absence of that range for that point in time. A Mel spectrogram has the same properties, but presents the frequencies on a logarithmic scale similar to the way the human ear works. High frequencies are squashed together since humans cannot distinguish between them in the first place, whereas the lower frequencies get to cover relatively more space in the image.

To compute a spectrogram, the audio fragment is divided into frames of 20-40ms, usually with a frame step size of 10ms to make them overlap. Each individual frame is then converted to a power spectrum, the distribution of power amongst the frequencies, using the Fourier transform

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i t \xi} dt$$

where $f(t)$ is the signal power at time t and $[\hat{f}(\xi)]^2$ is the total signal power for frequency ξ. The resulting spectrogram, consisting of all consecutive Fourier transforms, can then be mapped to he Mel scale by applying the Mel-spaced filterbank. Every column of pixels (i.e. every power spectrum), is multiplied by the 20-40 triangular filters that make up this filterbank. As shown in figure 3.3, the filter width and stride increase in size as the frequencies get higher, which collects the frequencies into bins according to the Mel-scale $M(f) = 1125 \cdot ln(1 + f/700)$. Finally, of each bin the logarithm is taken to compensate for the exponentially growing filter sizes, and those values are transformed using the discrete cosine transform to get the final image. The dimensions of the Mel spectrogram correspond to the amount of bins on the y-axis and the amount of frames on the x-axis. The values that make up the spectrogram, and therefore the spectrogram itself, are often referred to as the Mel frequency cepstrum coefficients, or MFCCs.

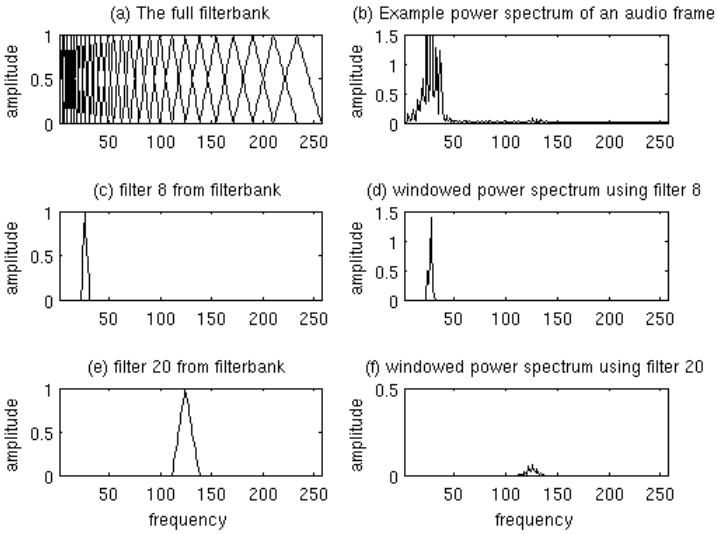


Figure 3.3: An example of a filterbank binning the signal (source: practicalcryptography.com)

3.3 Electronic Music

By its very nature, music genres are a very versatile and somewhat subjective type of target variable. The lack of an unquestioned truth, and the consequent types of ambiguity, call for a detailed discussion of some common misconceptions and a rationale for the asserted working definitions.

Electronic music, or simply electronic for short, is a term used to describe all music made primarily using digital instruments like keyboards or digital audio workstations (DAW). 'Dance' refers to the collection of electronic subgenres intended for the dance floor, thus excluding subgenres like ambient and chill out. However, since the majority of electronic subgenres are dance floor compatible, and because these are significantly more popular than those that are not, it is often used as a synonym for Electronic. The abbreviation 'EDM' stands

for electronic dance music, and thus appears to be an alternate name for the aforementioned collection. Yet the last decade this term has been used to refer specifically to arbitrary collections of a few popular subgenres, and skeptics have used it as a shorthand for all 'blockbuster' dance music when criticizing the integrity of the artists involved.

Electronic is different from other genres in that its defining elements, the instruments involved, apply to a lot of different styles. Rock, for instance, is not only defined by the instruments with which it is made, but also asserts a mood. The drums, electric guitar and bass guitar might just as well be used to make jazz, funk or blues music. One could therefore consider electronic music to be a realm of genres distinct from the 'analogue' genres, rather than a genre like any other. This could explain why electronic has significantly more subgenres than other genres. The amount of subgenres and the speed at which new ones are introduced is often subject to mockery. On the one hand, there is probably some truth to the notion of new subgenres actually being subcultural movements or blatant exaggerations of an artist's innovation within the genre, but on the other hand, many new subgenres are due to the development of a new feel, groove, or mood that distinguishes itself from its origins. In electronic music, a seemingly small change in rhythm can have a big impact on the groove (figure 3.4 shows the rhythms central to each genre). That being said, it is not always the sound of the music itself that determines the class label used to refer to a genre.

Subgenres in general can be defined by their geographical origins. Examples in electronic are French house and UK garage, which may go by different names when produced in other regions. Genres like brostep, the American flavor of dubstep, and gabber, a Dutch subgenre of hard dance, are associated with a region, yet exhibit stylistic characteristics of their own as well. Some subgenres are defined by subcultural origins. Acid jazz and deep house are notorious for being difficult to distinguish since the reported class labels depend largely on the classifier's musical paradigm. The usage of some class labels even depend on the attitude of the speaker towards the genre in question. As mentioned before, EDM has been used as a four-letter-word referring to popular trap, big room house, electro house, and any combinations of these, that are allegedly produced by dishonest artists. Their lack of integrity would manifest itself in the pursuit of commercial success above all else (also known as selling out), a disregard of artistic value, pandering, and the consequential lack of innovation and artistic identity. Another class label whose usage is dependent on people's attitude is deep house. As possibly the most ambiguous class label in today's music industry, the term 'deep house' is applied to many popular contemporary house releases. The sudden popularization of the term caused artists, labels and retailers to stretch the definition in an effort to profit from the hype. A more detailed discussion of the terminology is given in subsection 3.3.2.1.

Genre definition is only the first ambiguity in the classification process. Further confusion is caused by the ontological ambiguity. Examples are the so called 'umbrella genres', which are collections of several subgenres that do not necessarily fall under a single taxonomic node. Umbrella genres ignore the existing genre taxonomy and therefore classify music using different criteria. An example in electronic music is chill out. It includes subgenres like downtempo, trip hop, lofi hip hop, wonky, ambient and acid jazz, and even some subgenres from outside electronic. Other ontological ambiguities concern multiple inheritance, or the combining of genres.

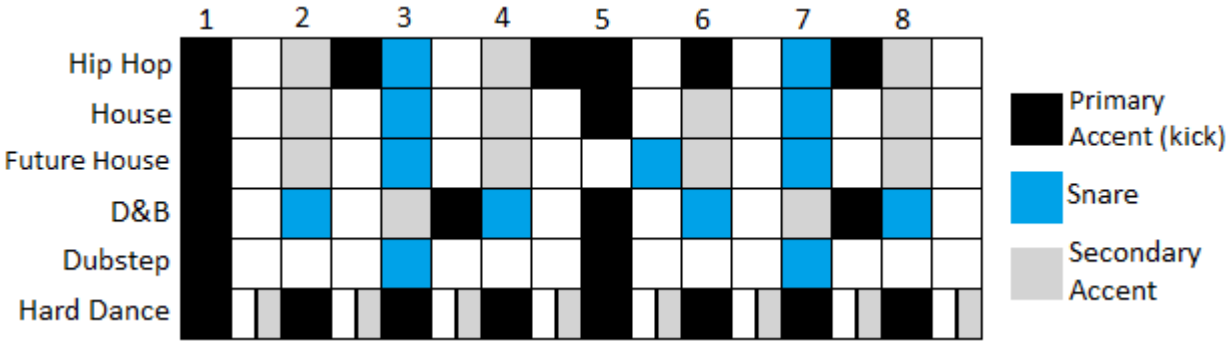


Figure 3.4: A schematic display of the rhythmic characteristics for each genre. The network models are expected to identify these patterns in the Mel spectrogram.

By combining two or more genres, artists can create a novel experience for the listener. Since this kind of innovation is celebrated among both artists and consumers, it has become quite a common practice. The frequency of this phenomenon has caused the taxonomical tree to be replaced by a directed acyclic graph. Once again, electronic music is the epitome of this behavior. In case of drumstep, chillstep or reggaestep, the uncreative naming convention makes it evident what genres are being combined (respectively drum & bass, chill out and reggae combined with dubstep, which in itself started as a combination of dub and 2-step garage). A less evident example is moombathon, a relatively new genre that combines house music with reggaeton. When the influences on a pre-existing genre does not result in a new genre altogether, the original genre can still change its course dramatically in the direction of its influencer, resulting in an equivalent outcome. Trap has existed since the 1990s as a subgenre of hip hop, but it only established its current sound during the rise of dubstep earlier this decade. Therefore, Trap cannot be considered a fusion between hip hop and dubstep, despite the sound proclaiming just that.

Contrary to musicians from analogue genres, few producers stick to a single subgenre for the entirety of their career, which causes personal styles to blend into other subgenres. As a result, musical genres overlap, fuse, and influence each other, causing them to reside on a multidimensional spectrum. This may well be the essence of what makes electronic music engaging to the listener, but it simultaneously obscures the process of classification. The following sections introduce the working definitions for the genres involved in the presented work, as well as their most prominent features and a rationale for their involvement. The sections include no technical details relevant to the classification, so if the composition of the dataset is of no concern, it is adviced to skip to chapter 4. The rhythmic characteristics of the electronic subgenres discussed below are summarized in figure 3.4.

3.3.1 Hip Hop

Although hip hop is officially part of the electronic music family, its independent subculture and massive popularity have placed it on the same taxonomic level as electronic music itself. Hip hop originating from this subculture often focuses on aspects like lyrics or rhyming flow, rather than rhythm. However, there is also hip

hop originating from within the electronic music subculture, which takes a completely different approach to the genre. Focusing on rhythm, as is commonplace in the electronic paradigm, this side of hip hop often lacks or disregard vocals. The infamous hip hop beat adds a phlegmatic vibe to any track that features it, which is typically applied for either of two purposes: apathy or relaxation. Two popular subgenres at the moment that embody these two purposes are trap, with its subgenre future bass, and lofi hip hop.
Trap originates from the late 1990s southern US. Like gangster rap and grime, the lyrics feature vile language and violent themes, creating an ominous mood. This ominosity is also evident in the music production through the use of minor harmonic scaled melodies, nervously rapid hi-hat segments and heavy basslines. The musical features became more prominent during the early 2010s due to the influence of dubstep, which caused trap to distance itself a little from hip hop subculture and join the electronic music scene.

Lofi hip hop is almost a direct opposite to trap. It features calm hip hop beats, a monotonous groove, and often incorporates modified jazz samples. This subgenre is currently up and coming, and so the naming convention is still undecided. People also refer to it as 'chillhop', 'jazzhop', 'study beats', or consider it to be an umbrella term for relaxing hip hop music in general.

Another up-and-coming genre is future bass. Officially, it is a subgenre of trap, but it stands in direct contrast to the ominosity and violence of its stylistic ancestor. The genre became mainstream in the mid-2010s by the likes of Chainsmokers and Marshmello. Like trap, it features the rapid hi-hats and prominent kicks, however it replaced the ominosity with soulful R&B chord progressions that modulate at every fill.

3.3.2 House

House emerged in Chicago in the 1980s as an extension of disco. It is characterized by a four-on-the-floor beat with with occasional off-beat hi-hats. The 1990s were the start of house as a mainstream genre when a subgenre called garage took Great Britain by storm. In subsequent years it spread throughout Europe, inciting the era of eurodance, and from that point on house has never left the charts. House music has blown up into many directions over the last 30 years and gained significant popularity. The genre is therefore divided into multiple major subgenres.

Progressive house is one of the first clearly distinguishable subgenres. As it emerged in the 1990s, producers took a lot of inspiration from alternative rock and trip hop in an effort to create a house subgenre for the alternative scene. The genre is characterized by constant steady build ups without the urge for drops or fills. The mood matches the time period and music scene, melancholic, psychedelic, or disinterested. Progressive house has held a steady popularity throughout the years, with producers like Deadmau5 and Kaskade who are among the most popular DJs in the world.

Electro house heads into the opposite direction and aims for energy and loudness. It makes heavy use of sawtooth synthesizers in simple, repetitive melodies. The origins are often traced back to Benni Benassi's 2002 single "Satisfaction". The subgenre knows many sides, some associating with EDM subculture like Steve Aoki, and some with bass culture like Feed Me and Knife Party. The the majority of electro house producers in bass culture belong to the subgenre known as complextro. Complextro evolved along with dubstep during the

bass culture hype earlier this decade. It features constantly switching synthesizers and sound effects to create a noisy and restless display. Many EDM associated electro house producers of the last few years belong to the subgenre called big room house. Big room emerged out of nowhere into mainstream popularity in 2013 when Martin Garrix released his single "Animals". The genre favors minimalistic drops in which the absence of synthesizers and the amount of reverb on the percussion creates a hollow feeling. Big room replaced the four-to-the-floor rhythm with something more similar to a hardstyle beat. However, the subgenre that has attracted most attention in recent years is deep house, or at least it is referred to as deep house.

3.3.2.1 Deep House

There has been a lot of confusion around the term 'deep house' the last couple of years. Its sudden jump to popularity since 2014 has flooded YouTube with 'deep house' playlists [12]. But more often than not, these playlists simply include popular house tracks, regardless of their actual subgenre. The two genres often referred to when talking about contemporary deep house are 'future house' and 'tropical house'. These are in fact considered to be subgenres of deep house, but could not be more different from the original sound. The original deep house that developed in late 1980s Chicago lacked all energy and positivity evident in these subgenres. Future house is heavily influenced by UK garage, primarily the rhythm. It creates a back beat by emphasizing the open hi-hats on the second- and fourth beat, rather than emphasizing the kick of the first- and third beat like traditional house music does. It also adds subtle percussive accents on several off-beats that incites the dance floor character. Melody-wise, future house often uses mechanical sound design, atonal pads and consecutive fifths. Tropical house is a slower-paced house genre aimed at festivals and summer parties. It often features horns or marimbas, and therefore has a more laid-back feel. Tropical house is very popular at the moment, and often the type of music people are looking for when they search for a deep house playlist.

3.3.3 Dubstep

Dubstep emerged from the UK rave scene in the early 2000s as a fusion of dub, electronic reggae made by Jamaican immigrants, and 2-step garage, the rhythmic underground part of UK garage. Its most characteristic feature is the slow half time tempo, closely followed by the deep sub-bass basslines. It first gained some popularity in 2009 when artists like Skream and Nero were featured in UK radio stations. In subsequent years, a new generation of artists surfaced that reinvented dubstep as an intensely aggressive genre. At the beginning of the new decade, Flux Pavilion released the infamous "I Can't Stop" and "Bass Cannon", both of which still pose as archetypes of this new era in dubstep. The tracks were comprised of the minimalistic dubstep half time rhythm, upon which a sharp and violent saw wave synthesizer created the melody. From this point on, dubstep enjoyed a sudden rise in popularity, bringing with it the new 'bass culture' music scene. The initial dubstep sound is often distanced from this new sound by classifying the latter as a subgenre called 'brostep'.

3.3.4 Drum & Bass

Drum & bass started as a genre called jungle and gradually morphed into a broader category of music that includes jungle as a subgenre. The invention of the sampler in the 1990s allowed DJs to record an excerpt of a different song, slice it into components, and reassemble the parts. The resulting productions featured fast, vibrant, ever-changing drum sections that gave jungle a restless and energetic character. As Simon Reynolds put it, "In jungle, the rhythm is the melody; the drum patterns are as hooky as the vocal samples or keyboard refrains" [33]. Typical drum & bass today is based on a standardized break beat consisting of a prominent snare on the second and fourth beat, a kick on the first beat, and the somewhat surprising kick in between the third and fourth beat. Although this rhythm is very frequently found in liquid funk, it is merely a stereotypical example of a drum & bass beat, and is for instance hardly ever found in neurofunk.

Liquid funk is a flavor of drum & bass that manages to combine the contradictory aspects of a fast, tense beat with relaxing, ambient synthesizers. Despite the relaxing sound, it is intended as dance floor music. High Contrast brought it to mainstream music in 2007 with his "Tough Guys Don't Dance" album. The moderation of the otherwise overly aggressive nature of drum & bass keeps it accessible to a wider audience than its sibling subgenres. Neurofunk, for instance, can be considered the direct opposite in that regard. It is an electronic adaptation of funk music, and as such, the traditional centerpieces of music like melody and chord progressions are of little importance. If there is any at all, the melody is simple and dissonant and only used to build up tension. The beat substitutes the melody as the headline of a track, frequently changing rhythmic accents during fills to surprise the listener, similar to funk break beats. Something unique to neuro are the so-called 'stabs', very rapid sawtooth sound effects meant to discomfort the listener. Neurofunk has always been an underground genre with little chance of mainstream success, but it has kept a very loyal following of consumers and artists.

3.3.5 Hard Dance

Hard dance is a collection of genres related to hardcore that do not have the sufficient characteristics to be considered part of the genre. Hardcore originates from the 90s rave era in the Netherlands. In a way, it is an electronic adaptation of Dutch folk music. The 2/4 rhythm accentuates every beat evenly and strongly with a saturated kick drum. Furthermore, the tempo is often between 170 and 200 beats per minute, and when these features are combined, an aggressive and sinister sound emerges. Most hardcore therefore also features minor chord progressions and dissonant melodies. The notable exception is happy hardcore, which got a lot of popularity in the late 90s and early 2000s, and still maintains a fair audience.

The reason for the introduction of 'hard dance' was the emergence of a related, but clearly different genre called hardstyle. During the mid-2000 some hardcore producers lowered the tempo to about 140-150 beats per minute. It appeared as a subsequent imperative to match the mood with the lowered tempo. The aggression was mostly replaced by uplifting chord progressions and motivating vocals, but it kept the violent 'screeches' and pumping rhythm. The moderation of the mood enabled its major mainstream success.

4 Approach

4.1 Dataset

The dataset ought to adhere to four principles: relevance, diversity, sufficient size, and non-ambiguity.

The main reason electronic music is chosen as a subject is because of its societal relevance. However, not all subgenres that technically belong to electronic are equally popular. To obtain this relevance, only the most popular contemporary subgenres are selected for the dataset. Popularity can be approached from the short term (relative popularity), or long term (absolute popularity). Long-term popularity varies less over the years, so results from two years ago are still applicable today. In 2015, EDM.com published the results of a questionnaire on favorite music genres, held within the US [12]. The first four genres of this list clearly distance themselves from the competition. These forerunners — trap, dubstep, electro house, and deep house — are therefore directly added to the dataset on grounds of long-term popularity along with their main subgenres — future bass, complextro, big room house, brostep, chillstep, future house, and tropical house. Places 5 to 12 — trance, drum & bass, progressive house, hardstyle, and techno (excluding aforementioned subgenres) — also managed to distance themselves from the last four. Of these five, trance and techno are considered to be dying out [13, 31], thus the other three are accepted, whereas these two are not. The last genre to be included before the composition is finalized is lofi hip hop, for the simple reason that its popularity has grown enormously over the past six months [40]. It is the new kid on the block and therefore the pinnacle among the short-term popularity genres.

As discussed before, datasets used in previous work often lacked diversity of artists. When certain artists influence a genre too much, the classifier will misinterpret the personal style of the artist as a feature of the genre. To overcome this vulnerability, the dataset is primarily comprised of compilation albums. By the very definition of the concept, the artist diversity in a compilation album is already ensured by its assembler. Furthermore, using multiple compilations by the same publisher extends this insurance across albums. Compilations are a common phenomenon in electronic music, and certain publishers have become renowned for focusing on specific genres, which serves the search for data perfectly. When the use of a single publisher for a genre is not possible, the intersections of the different publishers are manually checked for duplicate tracks. Remixes are not counted as duplicate tracks, since a remix intends to alter the original in such a way that it loses its previous identity.

Like the diversity, previous work often used datasets with dubious sizes. Although the norm in MGR is only 100 samples per class label, many studies used datasets far smaller, whereas other machine learning fields would collectively scoff at the marginality of such a norm. Going beyond 100 samples per genre is therefore

unnecessary, but of potentially substantial influence. However, genre labels unable to pass this threshold will be discarded from the dataset completely. Taking into account the taxonomical structure of the selected subgenres, the higher tier genres will inevitably have more samples than the lowest tiers. Exploring the influence of the dataset size is thus not only attainable by enriching the leaf genres, but can alternatively be simulated by increasing the level of abstraction.

All dataset samples are screened for ambiguity. This means that atypical tracks, assigned to a genre through desperation or in sheer compliance with a technical definition, are discarded from the dataset. The same applies to tracks with multiple genres. A frequently observed example in bass culture genres is a temporary switch to another genre for 8 to 16 bars. Some have solved this by classifying the track using a fusion genre, i.e. classifying dubstep tracks that switch to drum & bass as drumstep, yet this undermines the fusion genre as being a genre in its own right, asserting it to be a mere collection of tracks that can't be confidently assigned to either genre. To be safe, these tracks are disposed of as well.

Two different datasets are assembled, on of which is an extension of the other. This can give insight into the effect of dataset size and the amount of classes on the accuracy, as discussed in section 4.5.

4.2 Preprocessing

The preprocessing is a straightforward procedure. Given the root folder of the dataset, the system will loop through all subfolders and their mp3 files. First, the Mutagen module is used to extract meta information from the ID3 tags. The TCON frame containing the genre is extracted from the ID3 tag using `mutagen.id3.ID3`. Each track is tagged with only the lowest tier subgenre it belongs to. Secondly, the Librosa library is used to interpret the mp3 files. Each file is downsampled to 12000 kHz while calling `librosa.load` and only 29 seconds, conform to the approach by Choi [8], from the center of the track are used for further processing. Each excerpt is converted to a Mel spectrogram of 96 frequency bins, with a window size of 2048, and a stride of 256, using `librosa.feature.melspectrogram`. The process of creating a spectrogram is elaborated in section 3.2. Excerpts containing less than 1280 sound samples are discarded, and those with more are trimmed to exactly 1280 samples to keep input frame dimensions consistent. The track ID, genres, and the spectrograms as vectors, are then written to a CSV text file to save the progress. The genres are resolved to the required level of abstraction (explained in section 4.5), but before the data is actually fed into the network, the target variables are encoded to integers and converted to one-hot or k-hot encoding, depending on the amount of labels used in training. This is required to make the network output confidence scores rather than a single number, which would misrepresent the data, seeing as the label integers are not a continuum.

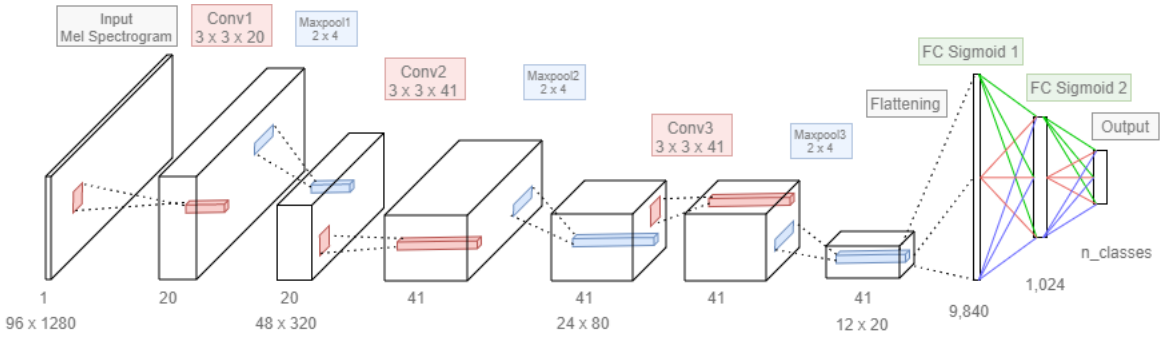


Figure 4.1: A graphic display of the small CNN

4.3 Models

The three presented networks are each based on the ones proposed by Keungwoo Choi [8]. As such, each of them receives single-channel two-dimensional Mel spectrograms as inputs, each applies dropout after every maxpool layer, each uses a sigmoid cross entropy loss function, and each trains using ADAM optimization. ADAM optimization is an alternative to the traditional gradient descent that takes into account previous gradients and controls the learning rate internally [21]. Contrary to Choi's models, the dimensions of the input frames and activation maps differ, as well as the learning rate and dropout rate since they were not disclosed in his paper.

4.3.1 Small CNN

This network is inspired by the k2c2 network described by Choi [8]. The original network achieved promising results for music genre recognition, and is therefore used to guide presented network architectures. The small CNN consists of three convolutional layers, each followed by a maxpool layer. The layers produce 20, 41, and 41 activation maps respectively. All convolution kernels are 3x3 with zero-padding, and all maxpools are 2x4, such that the 96x1280x1 input images are reduced to 12x20x41 after the last maxpool layer. Next, the resulting images are flattened and fed into a multilayer perceptron. The output is an array of dimension `n_classes`, which contains the probability per class as a floating point. The MLP contains one hidden layer of 1024 nodes, and a sigmoid activated cross entropy output layer. Sigmoid is chosen because the softmax function is incompatible with multi-label classification. The only difference between the first three layers of the k2c2 network and this network are the input and activation map dimensions. Choi starts with an input frame of 96x1366 and discards some edges in between layers to keep the frame dimensions divisible by the maxpool dimensions. To simplify this, the input image dimensions are a multiplication of the maxpool dimensions, which abolishes the need to reshape the image along the way.

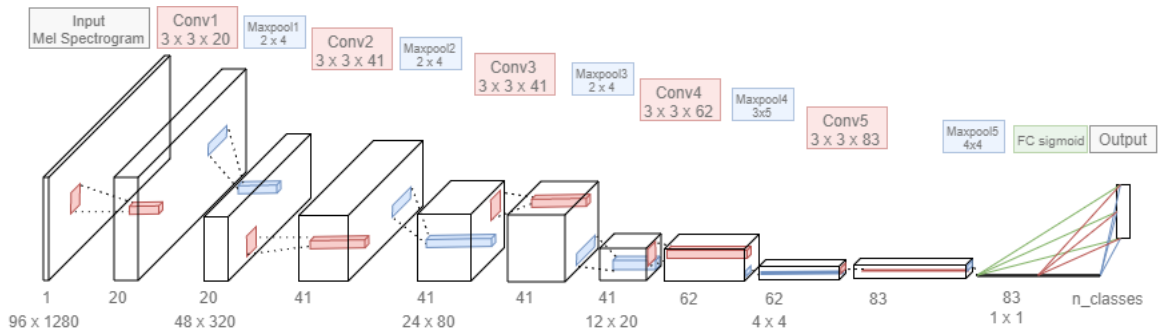


Figure 4.2: A graphic display of the k2c2 network

4.3.2 k2c2

The k2c2 algorithm is a complete reproduction of the fully convolutional network described by Choi [8] (with the exception of the frame dimensions and learning parameters described above). Like the small CNN, the first three convolutional layers and maxpool layers have the respective kernel dimensions of 3x3 and 2x4. Yet,these layers are then followed by two more 3x3 convolutional layers, the first of which is coupled with a 3x5 maxpool layer and the second with a 4x4 maxpool layer. This brings the output dimension down to single pixels, hence the name 'fully convolutional'. The last 2 convolutional layers produce 62 and 83 activation maps. The last 83 pixels are reduced to an array of dimension `n_classes` using the same fully connected sigmoid layer as the small CNN.

4.3.3 CRNN

This specific CRNN was also introduced in the same paper by Choi as a potential improvement on the k2c2 network. The difference in accuracy between the k2c2 and the CRNN were marginal in his evaluation, yet the hope is that electronic subgenres are more responding to the RNN layer. As Choi intended to keep the amount of gradients steady among his multiple architectures, the CRNN uses different amounts of activation maps, i.e. 30, 60, 60, and 60. The network starts with four 3x3 convolutional layers, each followed by a maxpool layer. The respective maxpool kernel dimensions are 2x2, 3x4, 4x4, and 4x4. Instead of the 3x4 kernel, Choi used a 3x3 kernel, but again due to the custom input dimensions, the x-axis is not divisible by 72. The result is therefore of dimension 60x1x10 (activation maps x frequency x time).

4.4 DAS-4

In order to limit the duration of the training procedure, the entire process is executed on the Distributed ASCI Supercomputer 4 (DAS-4) [4]. The DAS-4 is equipped with a collection of NVidia GTX Titan X graphics cards (harboring the GM204 GPU) running cuDNN, one of which is applied to accelerate the training procedures. It provides 4GB of RAM, and 40 GB of dataset storage space.

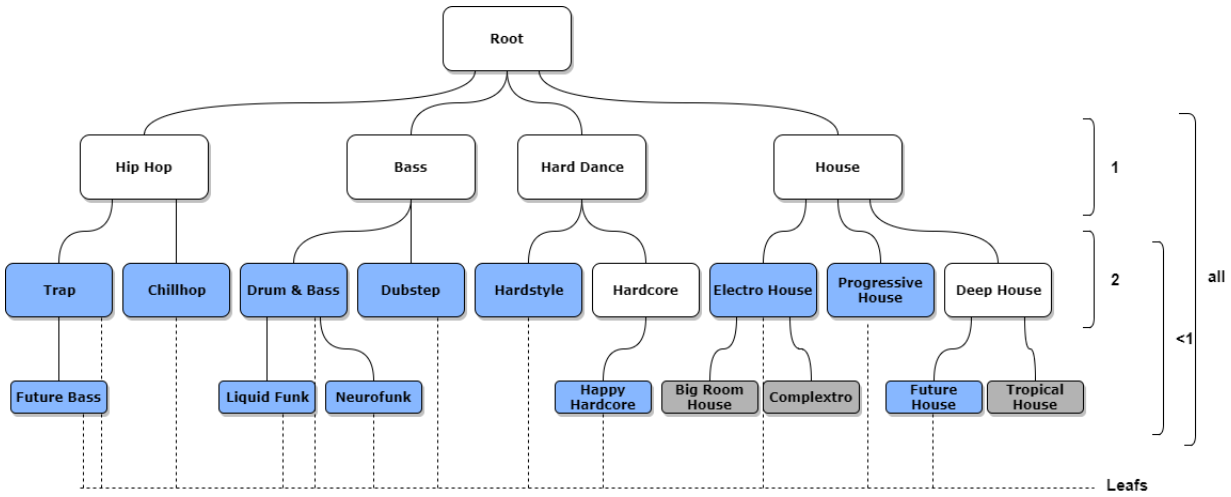


Figure 4.3: The genre hierarchy assumed for this project. Gray classes were not used as a distinct subgenre due to insufficient samples. Blue nodes have been used as labels to tag the dataset.

4.5 Evaluation

The evaluation explores the influence of five variables: the network, the dataset, the level of training abstraction, the learning rate and the dropout rate. These variables are used to determine the effect of subgenre targeting, dataset size and the number of target classes.

The focus of the research is on the influence of subgenre targeting. This is conceptualized into different levels of abstraction for the genre labels. As charted in figure 4.3, the dataset is organized as a tree. Five different levels of abstraction are identified: '1', '2', 'all', '◁1', and 'leafs'. Traditional MGR approaches train merely on the highest level of abstraction, taking into account only the main genres while disregarding the diversity within them. Because this dataset was tagged using the specific subgenres, the extra information about the taxonomy can be taken advantage of. The training and testing procedures can take place on different levels of abstraction, such that the model can be trained to predict abstraction 2 and tested on level 1 after abstracting the predictions back to their main genre. This could improve accuracy results since the differences in subgenres are not being averaged out but exploited for additional information. Another interesting option is multi-label training, providing multiple labels per sample. The extra information directly fed into the network could have positive influences on the accuracy as well. The evaluation therefore compares the results of models trained on each abstraction and tested on the highest abstraction. The testing abstraction keeps the difficulty of the classification task the same and enables insight into the effect of subgenre targeting.

Another inquiry is determining the influence of the number of classes and the size of the dataset. This is realized by the composition of two datasets, one being an extension of the other. The different results on each dataset indicates the effect of the amount of training samples as well as the amount of target classes. Section 4.1 provides details on the dataset composition.

Finally, the two different convolutional neural network models allow for a comparison of the two and a discussion on the use of CNNs in field of music genre recognition. A direct comparison to the algorithms made by Choi, however, is not possible, since they were evaluated using the area under the curve (AUC) of the receiver operator characteristic (ROC) (a composite evaluation measure incorporating true positives and false positives of binary classifiers) [8].

5 Results

5.1 Dataset

Two different datasets have been assembled, each of a different size and with a different amount of classes. Table 5.1 provides the exact meta information for each dataset. Three subgenres did not meet the threshold of 100 tracks. Therefore the big room house and complextro tracks were relabeled as plain electro house, whereas tropical house was completely discarded from the dataset.

Dataset Composition

			Dataset-1	Dataset-2
		uncompressed size	4.1GB	12.4GB
		preprocessed size	0.8GB	3.1GB
		amount of classes	4	12
main genre abs. 1	subgenre abs. 2	subsubgenre abs. 3	tracks	tracks
hip hop			173	297
	chillhop		94	100
	trap		79	75+122
		future bass		122
house			209	338
	electro house		109	109
	prog. house		100	100
	deep house			129
		future house		129
bass				370
	dubstep			100
	D&B			70+200
		liquid funk		100
		neurofunk		100
hard dance				346
	hardcore			226
		happy hardcore		226
	hardstyle			120
		total	382	1351

Figure 5.1: Information about the composition of each dataset. The left three columns represent the three layers of the dataset taxonomy. A sum in the rightmost column indicates the inclusion of tracks unassociated with any of the subsubgenres, making the subgenre a leaf as well.

5.2 Experiments

Three rounds of testing were conducted to narrow down the desired results.

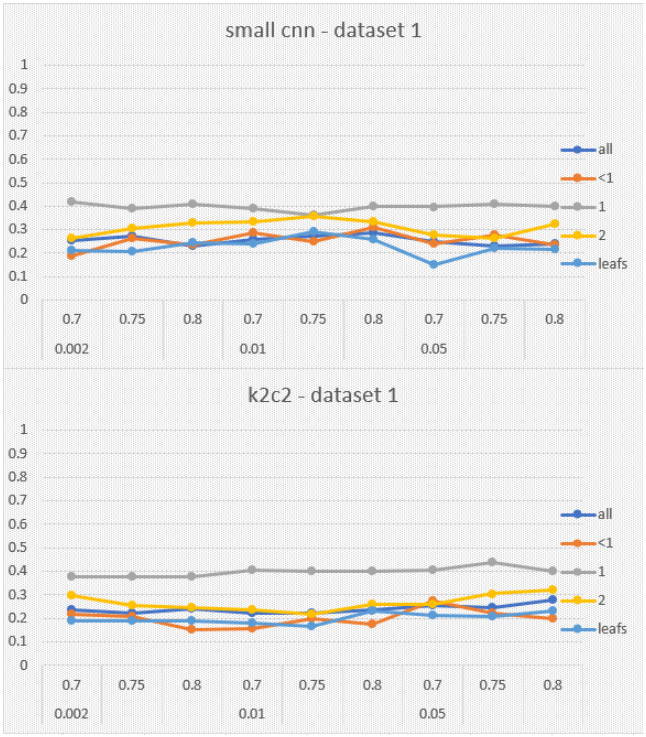
The first round of experiments aimed to find the most promising abstractions to be compared. Figure 5.2 shows the results. The accuracy results are each an average of five tests, in which each test in itself is already 5-fold cross validated. All tests took place on the highest abstraction. They were conducted in a hierarchical loop that initialized the dataset object anew for each level of abstraction, i.e. before the learning rate (α) and dropout (do) were being varied. However, since the dataset initialization is given a cross validation seed, all abstractions received the same convenient (or in convenient) dataset folding. It should be noted that, using a continuation of the same seed, each individual test gets its own fresh cross validation shuffle. The results for the abstractions are therefore based on 45 differently folded cross validated tests, but for each abstraction the exact same folds were used. The results consistently show abstraction 1 to have the best testing accuracy, followed by abstraction 2. The next round of testing only considered these two abstractions.

Round two featured the same test loop, but did not average over 5 tests. The 5-fold cross validation already

Round 1

small	CNN							
α	do	all	◁1	1	2	leafs	mean	best
.002	.70	.254	.190	.416	.262	.211	.267	.416
	.75	.272	.264	.391	.307	.205	.288	.391
	.80	.230	.237	.411	.330	.245	.291	.411
.010	.70	.258	.286	.389	.333	.241	.301	.389
	.75	.272	.248	.362	.359	.292	.307	.362
	.80	.287	.310	.401	.335	.258	.318	.401
.050	.70	.251	.238	.397	.277	.150	.263	.397
	.75	.231	.279	.408	.261	.219	.280	.408
	.80	.240	.234	.399	.324	.218	.283	.399
	mean	.270	.285	.408	.338	.229	.306	.408
	best	.287	.310	.416	.359	.292	.333	.416
k2c2								
.002	.70	.236	.217	.376	.299	.190	.264	.376
	.75	.223	.210	.379	.257	.191	.252	.379
	.80	.242	.150	.377	.246	.188	.241	.377
.010	.70	.221	.154	.407	.237	.179	.240	.407
	.75	.220	.198	.399	.218	.166	.240	.399
	.80	.236	.174	.400	.259	.231	.260	.400
.050	.70	.257	.274	.405	.260	.213	.282	.405
	.75	.244	.222	.437	.304	.209	.283	.437
	.80	.278	.197	.401	.319	.231	.285	.401
	mean	.252	.230	.407	.293	.218	.280	.407
	best	.277	.274	.437	.319	.231	.308	.437

(a) The accuracy for all models and abstractions on the first dataset



(b) The graphed results of round one. Abstractions 1 and 2 are the clear winners.

Figure 5.2: The results of round one, meant to determine the most promising abstractions. All scores are the average of five separate 5-fold cross validated tests, and the 0R for each test is 0.547.

made the results sufficiently resistant to luck and thus equally reliable for evaluation. Figure 5.3 presents the results. This round aimed to optimize both tasks on dataset-1, making the accuracy results more representative of their predictive quality.

The learning rates and dropout rates being tested in round 2 were established in a small prior test with both models on the first dataset. The values chosen for these prior tests were ten equally spaced values around the learning rate used in the Tensorflow MNIST example, and ten equally spaced values around the dropout used by Choi. The three overall best values were selected for further testing; .025, .05, and .075 for the learning rate, and .4, .5, and .6 for the dropout respectively. Note that both networks use ADAM optimization to control the learning rate [21], and so the presented values are only the starting values for the optimizer.

The tests in round three on dataset 2 were different from those on the first dataset due to an unanticipated shortage of memory on the DAS-4 (see chapter 7). To minimize the amount of training epochs, the parameters were instead chosen to optimize accuracy for each individual test run. Acknowledging the potentially different needs among the four tasks, all tests were guided by previous results to find the optimal parameter configurations. Additionally, the optimal parameters for each task were also tested on the other 3 tasks for more reliable comparison. The results are shown in figure 5.4.

To interpret the accuracy results, one ought to take into account the zero rule (0R). This is the baseline accuracy for any asymmetrical classification task. The random baseline, i.e. the accuracy of a random classifier, does not apply to asymmetrical classification since the classes are of different sizes. The zero rule, however, is the accuracy of a hypothetical model that always picks the most frequent label seen during testing. To compute this, the most frequently occurring label in the dataset is identified, then for that label the testing abstraction is taken. The size of that genre relative to the whole dataset is the zero rule. In this case, the zero rules for both datasets are the same for each of their abstractions. For dataset 1, the most frequent test label is either electro house or house, but since electro house is abstracted to house, all zero rules are $109/382 = .5471$. For dataset 2, the same principle applies to bass and drum & bass, making its zero rules $370/1351 = .2739$.

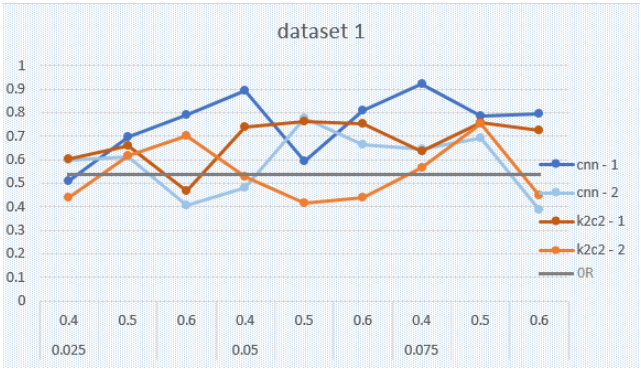
5.2.1 Subgenre Targeting

Both rounds of testing on the first datset tell us that the best results are achieved by training on the highest abstraction for either model and any combination of parameters. The only parameters more suited for subgenre targeting were $\alpha = .025$ and $do = .4$ when trained on the k2c2 model. The constant second place goes to the second abstraction, and the last place to the leafs. Training on the 'all' abstraction produces the most stable results with the smallest variance, and the least stable abstraction to trian on is that of the leafs. Multi-label classification, the 'all' and '$\lhd$1' abstraction, did not improve regular classification. However, it is notable that the 'leafs' abstraction scored even worse than the both of them. In round 3, the best accuracy is achieved by targeting level 2 subgenres using the small CNN, ranking 13% higher than traditional targeting using the same network. The k2c2 did not improve by using subgenre targeting.

Round 2

α	do	cnn 1	cnn 2	k2c2 1	k2c2 2	mean	best
.025	.4	.510	.597	.603	.440	.536	.603
	.5	.697	.611	.660	.619	.647	.697
	.6	.791	.407	.468	.700	.592	.791
.05	.4	.892	.480	.741	.529	.661	.892
	.5	.594	.777	.763	.413	.637	.777
	.6	.810	.662	.754	.439	.666	.810
.075	.4	.924	.644	.637	.564	.692	.924
	.5	.787	.693	.757	.752	.747	.787
	.6	.795	.385	.725	.446	.590	.795
	mean	.756	.584	.679	.545	.641	.756
	best	.924	.777	.763	.752	.804	.924

(a) The accuracy scores on dataset 1 for the two most promising abstractions and both models.



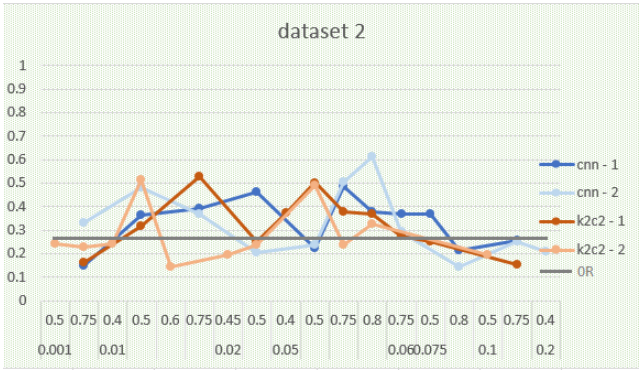
(b) The graphed results of round two. The small CNN, trained on abstraction 1, achieved the best accuracy.

Figure 5.3: The results of round two. All tests were 5-fold cross validated, and the 0R for each test is 0.547.

Round 3

α	do	cnn 1	cnn 2	k2c2 1	k2c2 2	mean	best
.001	.5				.243		
	.75	.147	.333	.161	.228	.217	.333
.01	.4				.240		
	.5	.363	.480	.319	.512	.419	.512
	.6				.144		
	.75	.394	.369	.530		.431	.530
.02	.45				.194		
	.5	.461	.205	.253	.238	.289	.461
.05	.4			.373			
	.5	.225	.235	.502	.492	.364	.502
	.75	.488	.504	.376	.237	.401	.504
	.8	.377	.611	.367	.324	.420	.611
.06	.75	.367	.295	.273		.312	.367
.075	.5	.368		.253		.311	.368
	.8	.215	.142			.179	.215
.1	.5				.194		
	.75	.254	.252	.154		.220	.254
.2	.4		.210				
	mean	.333	.331	.324	.277	.316	.333
	best	.488	.611	.530	.512	.535	.611

(a) The accuracy scores on dataset 2 for the two most promising abstractions and both models



(b) The graphed results of round three. The small CNN, trained on abstraction 2, achieved the best accuracy.

Figure 5.4: The results of round three. All tests were 5-fold cross validated, and the 0R for each test is .274.

5.2.2 Networks

Looking at the tests on the first dataset, the overall best parameter settings for the small cnn are $\alpha = .075$ and $do = .4$, which yielded the best result of all tests, and for the k2c2 $\alpha = .05$ and $do = .5$. The dropout seems to have least effect on the accuracy, the learning rate some, and the training abstraction most. The k2c2 algorithm scored best in round 1, with both a best accuracy and a total mean of about 2% higher. But in round 2, the CNN scored a best accuracy 16% higher-, and a mean accuracy 7% higher than the k2c2 for abstraction 1. For abstraction two, the CNN still prevails but only with respectively 2.5% and 4%. In round 3, the CNN achieved the overall best accuracy of 60.1% when its parameters were set to $\alpha = .05$ and $do = .8$.

5.3 Datasets

Due to the difference in classification difficulty, the best results for both datasets should first be mapped to match their zero rule before they can be compared. For dataset 1, the best result of 92.4% was achieved with a 0R of 54.7%, which makes the normalized result $(.924 - .547)/(1 - .547) = .832$. For dataset 2, the best result of 61.1% was achieved with a 0R of 27.4%, thus the normalized result is $(.611 - .274)/(1 - .274) = .464$. The tests in round 2 on dataset 1 therefore resulted in a considerably higher accuracy than the tests in round 3 on dataset 2.

6 Conclusion

The targeting of subgenres during training achieved only moderate improvements in classification accuracy compared to targeting the main genres. Its effects appear to be solely visible when using the small CNN on a large dataset. There might be two conflicting phenomena at work that explain this observation. Firstly, what subgenre targeting intends to do is to capture more information about a genre, namely the characteristics of its subgenres. Getting to know each genre better might require more data before it can contribute to the accuracy. Secondly, under the surface, subgenre targeting sabotages its own trianing process by having to attribute more features to multiple classes. The weights associated with those features that are central to a genre are essentially shared across the subgenres, lowering their influence in the network. This would improve overall accuracy when tested on the same abstraction, but the ability to differentiate between different subgenres within the same genre does not justify the loss of association between the weights and the main genre. This second hypothetical would have more influence the more subgenres there are relative to the amount of main genres. Compared to the first dataset, which has two subgenres per main genre, the second dataset contains 9 subgenres for 4 main genres. This extra singleton did not outweight the increase in dataset size, supposedly resulting in better subgenre targeting results for the second dataset.

Contrary to initial assumptions, the small CNN does seem to outperform its big brother. Although it was intended as a k2c2 cut short by two layers, it appears that the multilayer perceptron at the end has had a positive effect on classification results. However, when counting the amount of parameters for each model, the multilayer perceptron appears to have a lot more than the last two layers of the k2c2, which explains the observations. The MLP maps 41 12x20 activation maps to 1024 hidden nodes, and those hidden nodes to the output layer, which amounts to $41*12*20*1024+1024*n \approx 10^7$. The last two convolutional layers have 62 and 83 3x3 kernels, and map 83 nodes to the output layer, resulting in $3*3*41*62+3*3*62*83+83*n \approx 69*10^3$. Coming back to the subgenre targeting, the additional specifics on the subgenres require more parameters to harbor that information. The amount of parameters in the 'small' CNN can explain its superiority. The 92.4% accuracy achieved by this model is a token of potential, but binary classification is a long way from the typical 10 class classification.

Although the two datasets can be compared in terms of classification accuracy, the cause of that difference can be attributed to either the number of classes or the total amount of samples. Because there was no control group for either of these variables, it is troublesome to draw any definite conclusions about them. However, what has become clear is that the first dataset achieved better results than the second. The increase in size did not outweigh the increase in classes, despite the increased average amount of samples per class. Therefore, either the ratio of samples per genre should be higher, or the typical use of small datasets in MGR is justified.

7 Discussion

Contrary to the research by Choi, the experiments performed in this paper were not memory controlled. It could be considered an unfair advantage if one algorithm is allowed to use more system resources than the others. Furthermore, the choice of Python as a program language disabled the algorithms from being memory optimized, as Python handles its memory allocation and garbage collection internally. This might as well have been the reason the second dataset could not be trained on the DAS-4. The GPU node provides 4GB of RAM, and the second dataset amounts to 3.1GB after preprocessing. It should therefore have been possible to fit the entire dataset in memory, yet a memory exhaustion exception prevented the second dataset form being used on the DAS-4 completely. All subsequent tests were performed on a separate desktop computer with 16GB of RAM. There it became evident that running the small CNN on dataset 2 required 7GB of RAM, whereas the k2c2 even peaked at 11GB of RAM usage. This new system did not provide GPU acceleration, and so the parameters for the tests had to be selectively chosen. The initial plan was to test each combination of training- and testing abstraction, rather than testing on abstraction 1 only. Specifically the evaluation of training and testing on abstraction 2 could have been useful in comparing the amount of classes. Besides memory control, the experiments were also not parameter controlled. As it became evident in the conclusion, the 'small' CNN posessed over 100 times more parameters than the k2c2 model. This too could very well prevent a fair comparison between the algorithms.

All accuracy scores reported for round 1 of testing are below the zero rule of 54.71%, which brings about the question whether these results are justified for comparison at all. It would require a serious effort to optimize the parameters for each abstraction and model, and such a thorough evaluation is beyond the scope of this project. It ought be noted that the conclusion of the project is based on the assumption that these results were sufficiently reliable to determine the two best levels of abstraction.

One of the things to be taken into account when replicating this project is to unit test each function before assuming their functionality. Time did not allow for proper unit testing, and the presence of overlooked bugs is therefore possible and probable. The results might be compromised if their computation was faulty. Although no known bugs appeared to influence the accuracy, the lack of proper unit testing alludes to the presence of bugs in the project. A known bug that has not been tackled relates the the CUDA memory allocation. Every first training session fails due to a lack of system resources, yet all subsequent sessions run seamlessly. Because the failed sessions are discarded, this bug did not influence the accuracy.

7.1 Future Work

First and foremost, any future work into subgenre targeting is encouraged. Although it showed potential, no concrete conclusions could be drawn from these results. Any follow up study ought to improve on the software engineering principles neglected by this project. The experiments need to be controlled for network parameters, target classes, and dataset size. The use of more samples and bigger networks should increase the effect of subgenre targeting if the trends found in these results are true. Frankly, two options for dataset size and network parameters cannot establish a trend as such. Future research could therefore test their effects on a more continuous scale. It might also be valuable to pop the hood of the subgenre predictions during testing.

The project held the intention of exploring a convolutional recurrent neural network (CRNN), since these are the best state of the art networks in MGR. However, the mere 12 weeks available did not permit this, considering the lack of prior experience with Tensorflow. Future work would therefore first and foremost be advised to implement a CRNN and put it to the test.

This project was based on the convenient assumption that a song exhibits most character in the temporal middle. However, the dataset assures nothing of the like. Future research on electronic music could invest time developing a function that can scan a track and locate the drop. With such a function, the sample can be taken from directly after the drop, which is often the most characteristic part for those genres. Alternatively, a network ending in a recurrent layer can provide a continuous output throughout the time domain. This can normalize differences between parts of the track, continually report new labels throughout, or only report a label when its confidence socre is above a certain threshold. Even the sampling of multiple excerpts using a CNN and applying a majority vote could make the results more consistent and reliable.

The criticism expressed by Sturm on the use of simple classification in MGR is well founded [38]. For electronic music specifically, genres are defined according to only a few variables, e.g. rhythm, tempo, chord progression (mood), melody (if any). If all three are classified separately, the resulting genre label could be reconstructed from the component labels. This is another way to more specifically tell a network what to focus on, similar to what the subgenre targeting intends to do. The intermittent results could be tested individually for more reliable accuracy results. Another way to replace classification which has already been applied by previous work is to place the genres on a spectrum in n-dimensional space and evaluate the angle between the prediction vector and genre. This acknowledges the continuity of music genres, but is heavily dependent on the conversion from genre label to vector, which in itself is a subjective task.

Something this project lacked was an evaluative look inside the classification process to identify frequent mistakes. Knowing exactly where and when the algorithm fails uncovers the cause. Through lack of such an evaluation no conclusions can be drawn about consistently different accuracy results between subgenres. Future work could include this, and ideally apply it on the two proposed datasets. Another way to look into the mechanics of the classification is to deconvolute the networks and see to what patterns they ascribe their results.

Bibliography

[1] Alex Alexandridis, Eva Chondrodima, and Georgia Paivana Marios Stogiannos Elias Zois Haralambos Sarimveis. Music genre classification using radial basis function networks and particle swarm optimization. pages 35–40. IEEE, September 2014.

[2] Abel Alvarado. It's a $6.2 billion industry. but how did electronic dance music get so popular? `http://edition.cnn.com/2014/12/18/world/how-did-edm-get-so-popular/index.html`, 2015. Accessed: 02-07-2017.

[3] Iasonas Antonopoulos, Aggelos Pikrakis, Sergios Theodoridis, Olmo Cornelis, Dirk Moelants, and Marc Leman. Music retrieval by rhythmic similarity applied on greek and african traditional music. pages 297–300. ISMIR, Austrian Computer Society, September 2007.

[4] Henri Bal, Dick Epema, Cees de Laat, Rob van Nieuwpoort, Frank Seinstra John Romein, Cees Snoek, and Harry Wijshoff. A medium-scale distributed system for computer science research: Infrastructure for the long term. 49:54–63, May 2016.

[5] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. 2, October 2011.

[6] Austin C. Chen. 'automatic classification of electronic music and speech/music audio content', msc thesis. 2014.

[7] Kyunghyung Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. October 2014.

[8] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Convolutional recurrent neural networks for music classification. December 2016.

[9] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. December 2014.

[10] Oliveira L.S. Costa, Y.M. and C.N Silla. An evaluation of convolutional neural networks for music classification using spectrograms. *J Intell Inf Syst*, 52:28–38, 2017.

[11] Jia Dai, Shan Liang, Wei Xue, Chongjia Ni, , and Wenju Liu. Long short-term memory recurrent neural network based segment features for music genre classification. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5. IEEE, 2016.

[12] EDM.com. We've uncovered the most popular edm genre in the us [study]. http://edm.com/articles/2015/10/28/most-popular-edm-genre-us, 2015. Accessed: 17-06-2017.

[13] Larry Fitzmaurice. It's 2016: Are house and techno thriving or stagnating? https://thump.vice.com/en_us/article/its-2016-are-house-and-techno-thriving-or-stagnating, 2016. Accessed: 17-06-2017.

[14] Juliano Henrique Foleiss and Tiago Fernandes Tavares. A spectral bandwise feature-based system for the mirex 2016 train/test task. In *17th International Society for Music Information Retrieval Conference*. ISMIR, August 2016.

[15] Anshuman Goel, Sarfaraz Masood, Hohd Sheezan, and Aadam Saleem. Genre classification of songs using neural network. pages 285–289. IEEE, September 2014.

[16] Shahram Golzari, Shyamala Doraisamy, Md Nasir Sulaiman, and Nur Izura Udzir. A hybrid approach to traditional malay music genre classification: Combining feature selection and artificial immune recognition system. *International Symposium on Information Technology*, 2:1–6, 2008.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[18] Jeremy Irvin, Elliot Chartock, and Nadav Hollander. Recurrent neural networks with attention for genre classification. 2016.

[19] ISMIR. Our partners. https://wp.nyu.edu/ismir2016/industry-partners/our-partners/, 2016. Accessed: 15-06-2017.

[20] Alexey Grigoryevich Ivakhnenko and Valentin Grigor'evic Lapa. American Elsevier, New York, 1967.

[21] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015.

[22] Priit Kirss. 'audio based genre classification of electronic music', mmt thesis. 2007.

[23] Qiuqiang Kong, Xiaohui Feng, and Yanxiong Li. Music genre classification using convolutional neural network. In Wang et al. [43].

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[25] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

[26] Yann LeCun, Léon Bottou, and Yoshua Bengio andPatrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998.

[27] Tom LH. Li, Antoni B. Chan, and Andy HW. Chun. Automatic musical pattern feature extraction using convolutional neural network. In S. I. Ao, Oscar Castillo, Craig Douglas, David Dagan Feng, and Jeong-A Lee, editors, *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, pages 546–550. IMECS, Newswood Limited, March 2010.

[28] Thomas Lidy and Alexander Schindler. Parallel convolutional neural networks for music genre and mood classification. In *17th International Society for Music Information Retrieval Conference*. ISMIR, August 2016.

[29] Siva Sankalp P., Trinayan Baruah, Shlesh Tiwari, and Sankar Ganesh S. Intelligent classification of electronic music. pages 31–35. IEEE, June 2014.

[30] Maria Panteli, Bruno Rocha, Niels Bogaards, and Aline Honingh. A model for rhythm and timbre similarity in electronic dance music. *Musicae Scientiae*, June 2016.

[31] RD. Beatport sales chart: Dubstep is dead. `https://producerdj.com/2014/08/beatport-sales-chart-dubstep-is-dead/`, 2014. Accessed: 17-06-2017.

[32] Simon Reynolds. Picador, London, UK, 2012.

[33] Simon Reynolds. Picador, London, UK, 2012.

[34] Simon Reynolds. How rave music conquered america. `https://www.theguardian.com/music/2012/aug/02/how-rave-music-conquered-america`, 2012. Accessed: 17-06-2017.

[35] Jüürgen Schmidhuber. Learning complex, extended sequences using the principle of history compression. *Neural Computation*, 4:234–242, 1992.

[36] Siddharth Sigtia, Emmanouil Benetos, Srikanth Cherla, Tillman Weyde, Artur S. d'Avila Garcez, and Simon Dixon. An rnn-based music language model for improving automatic music transcription. In Wang et al. [43], pages 53–58.

[37] Bob L. Sturm. An analysis of the gtzan music genre dataset. In *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, pages 7–12. ACM, 2012.

[38] Bob L. Sturm. Classification accuracy is not enough: On the evaluation of music genre recognition systems. *Journal of Intelligent Information Systems*, 41:371–406, 2013.

[39] Bob L. Sturm. *Adaptive Multimedia Retrieval: Semantics, Context, and Adaptation*, volume 8382, chapter A Survey of Evaluation in Music Genre Recognition, pages 29–66. Springer International Publishing Switzerland, Cham, Switzerland, 1 edition, 2014.

[40] Google Trends. Lofi hip hop interest over time. `https://trends.google.de/trends/explore?q=lofi%20hiphop`, 2017. Accessed: 02-07-2017.

[41] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. In *IEEE Transactions on speech and audio processing*, volume 10, pages 293–302. IEEE, July 2002.

[42] Benedikt S. Vogler, Amir Othman, and Günter Schatter. Music genre recognition. 2016.

[43] Hsin-Min Wang, Yi-Hsuan Yang, and Jin Ha Lee, editors. *Proceedings of the 15th Conference of the International Society for Music Information Retrieval (ISMIR 2014)*. ISMIR, October 2014.

[44] Ming-Ju Wu and Jyh-Shing Roger Jang. Combining acoustic and multilevel visual features for music genre classification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 12, 2015.

[45] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. October 2016.