Transfer Learning in Neural Networks: An Experience Report

Mark Shtern
York University
4700 Keele Street
Toronto, ON
mark@cse.yorku.ca

Rabia Ejaz
York University
4700 Keele Street
Toronto, ON
rabia93@my.yorku.ca

Vassilios Tzerpos
York University
4700 Keele Street
Toronto, ON
bil@cse.yorku.ca

ABSTRACT

Perhaps the most important characteristic of deep neural networks is their ability to discover and extract the necessary features for a particular machine learning task from a raw input representation. This requires a significant time commitment, both in terms of assembling the training dataset, and training the neural network. Reusing the knowledge inherent in a trained neural network for a machine learning task in a related domain can provide significant improvements in terms of the time required to complete the task.

In this paper, we present our experience with such a transfer learning situation. We reuse a neural network that was trained on a real world image dataset, for the task of classifying music in terms of genre, instrumentation, composer etc. (audio files are converted to spectrograms for this purpose). Even though the image and music domains are not directly related, our experiments show that features extracted to recognize images allow for high accuracy in many music classification tasks.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; *Supervised learning by classification*;

KEYWORDS

deep learning, transfer learning, music classification

ACM Reference format:
Mark Shtern, Rabia Ejaz, and Vassilios Tzerpos. 2017. Transfer Learning in Neural Networks: An Experience Report. In *Proceedings of CASCON, Markham, Ontario, Canada, November 2017 (CASCON 2017),* 10 pages.

1 INTRODUCTION

The decomposition of a set of objects into meaningful classes is a problem that has attracted attention since the beginning of civilization. Humans are intuitively good at determining the relevant features that can help them to effectively cluster such a set. For more advanced categorization tasks, supervised and unsupervised methods have been developed to formalize the process of grouping similar objects together, and improve its accuracy. Recently, the advent of deep learning has produced highly effective approaches for the traditionally hard (for computers) problem of image classification [8, 24]. However, the accuracy of these approaches relies on the availability of powerful specialized hardware and large labeled datasets.

When the number of available labeled data is not sufficient to effectively train a neural network for a particular classification task, it is still possible to use a network that has been trained for a task in a related domain. For example, for a task where obtaining labeled real world data is not easy, it may be possible to train a model using simulated data and fine-tune it using the real data that is available. Such an approach, where knowledge gained in one domain is transferred to a new domain, is referred to as *transfer learning*.

In a transfer learning scenario, the labeled data of the new domain may be used to re-train the whole network, part of it, or only its output layer. In this paper, we concentrate on the latter option. We refer to such an approach as *black box transfer learning*, since we treat the trained network as a black box that receives the labeled data as input and provides a vector of outputs to be used as inputs to the last layer.

The goal of this paper is to investigate whether state of the art image classifiers can be employed to solve classification problems in the music domain. For this purpose, we utilize several audio datasets that we created specifically for this paper, as well as existing datasets that have been used by other researchers. We convert the audio clips in these datasets to spectrograms that are provided as input to image classifiers that have been pre-trained with real world images, such as images of everyday objects or animals. We treat the image classifier as a black box and retrain only its output layer for each specific classification experiment.

Our results show high accuracy for several of the classification tasks we experimented with. In many cases, our transfer learning approach performed as well as a special-purpose approach that used direct domain knowledge by extracting features directly from the audio (the special-purpose approach outperformed our approach in other experiments). This shows that knowledge gained in the real world image domain can be sufficiently transferred to the spectrogram domain (and by extension to the audio domain) making black box transfer learning a feasible and efficient approach.

The structure of the remainder of this paper is as follows: Section 2 presents the necessary background in audio signals and spectrograms, while Section 3 discusses related work in transfer learning and music classification. The datasets we used, as well as the setup for our experiments is presented in Section 4. We discuss the results of these experiments in Section 5. Follow-up experiments and the related discussion is presented in Section 6. Finally, Section 7 concludes the paper and presents future work.

CASCON 2017, November 2017, Markham, Ontario, Canada

Figure 1: A waveform of a birdsong showing 5 bird calls [2]

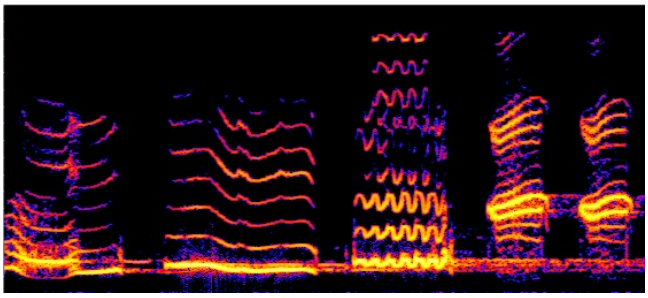


Figure 2: A spectrogram of a birdsong showing the distinct harmonic spectra of 4 different bird calls [2]

2 BACKGROUND

Digital audio signals are stored using the pulse-code modulation (PCM) method (.wav files contain PCM data). While this format is great for storage and reproduction, it only stores the amplitude of the signal at every sampling point. As a result, its visual representation, called a waveform (see Figure 1), only provides information about the loudness of the signal at any given time. For most classification tasks however, one needs to know the prominent frequencies in the signal.

For this reason, the visual representation used for classification purposes is that of a spectrogram. In a spectrogram, the x-axis is still time, but the y-axis is frequency instead of loudness. The larger the intensity of a pixel, the larger the magnitude of that frequency in the signal. Colour can be added to a spectrogram to denote phase information, as is the case in Figure 2. In this paper, we ignore phase information as the human ear is insensitive to phase. [1]

Figure 2 presents the spectrogram that corresponds to the waveform in Figure 1. It also showcases the fact that real world signals have multiple frequencies present at any given time. In fact, even if a single note is played on a piano, the resulting signal will contain several frequencies, typically multiples of the lowest present frequency, called the fundamental frequency. The distribution and magnitudes of these frequencies is what gives each instrument its distinct sound, its *timbre*.

Our conjecture is that a neural network that has been trained with real world images, and has therefore learned to detect edges and contours in them, should be able to distinguish the timbre of different audio sources, such as various instruments, with high accuracy if given a spectrogram as input. That is because the timbre of a sound, i.e. the distribution of overlapping frequencies, creates distinct patterns in a spectrogram. These patterns are, of course, harder to distinguish in a complex piece of music featuring several instruments playing at the same time, but our results show that our classifier still manages to classify with high accuracy.

[1] Phase can have a significant effect when combining two or more signals. This situation does not appear in our work.

It is also important to note that the visual representation we have chosen for this paper will affect the accuracy of some classification tasks more than others. For example, if the classes in a particular classification task represent different levels of loudness, then maybe the waveform would be a more appropriate representation. More importantly, if the distinguishing feature between the classes is timing as opposed to timbre, one should expect accuracy to degrade.

To illustrate this point, consider the spectrograms is Figure 3. MIDI guitar was used to create the audio clip that corresponds to the top two spectrograms, while MIDI piano was used for the bottom two. The distinct timbre of each instrument is apparent (for all clips a C major chord was repeated a few times), so one would expect our classifier to perform well in an instrument classification task.

On the other hand, the two spectrograms on the left correspond to an audio clip where the C major chord is repeated in $\frac{3}{4}$ time (every third repetition of the timbre pattern is slightly brighter[2]. Similarly, the two spectrograms on the right correspond to $\frac{4}{4}$ time (every fourth repetition is brighter). While it is possible to see the distinction, one would expect that a neural network trained with real world images would be more likely to learn to identify the timbre pattern rather than the timing pattern.

In the next section, we present related work in transfer learning and music classification before presenting our experimental setup in Section 4.

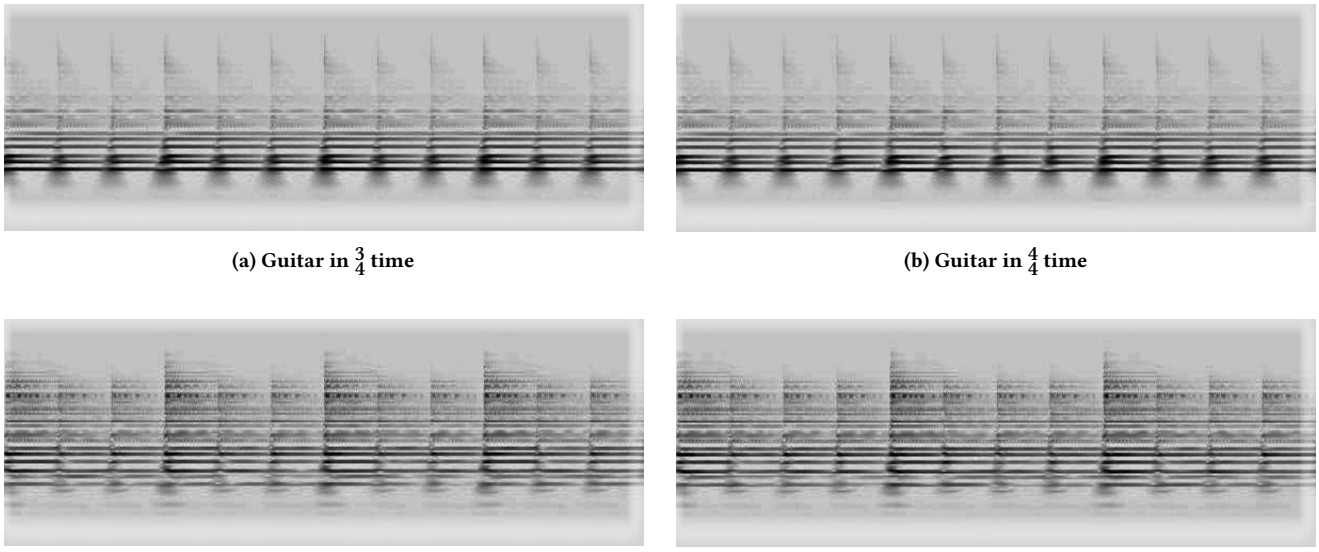
3 RELATED WORK

Transfer learning is the transfer of knowledge and skills between domains. It uses knowledge gained in a source task to improve a related target task. In classification tasks, it is often not possible to collect the needed training data from scratch. This limited supply of labeled training data is the inspiration for transfer learning. This technique can significantly reduce the number of trainable parameters in the target domain by transferring already trained weights.

In deep neural networks, transfer learning is done by reusing the pre-trained models and making adjustments per your own dataset. This is done by removing the last layer of the network and adding one or more new layers to the model (depending on your task). The pre-trained model is then fine-tuned to solve the new problem. Since the pre-trained network is assumed to be already trained in a meaningful fashion, the weights of existing layers are not expected to need significant re-training to converge (or may not be re-trained at all as is the case with black box transfer learning). In this way, the number of parameters that must be learned decreases significantly, as does the amount of training necessary.

There are many examples where transfer learning is truly helpful. For instance, transfer learning is used in human activity recognition in smartphones. The aim of activity recognition is to recognize common human activities in real life such as monitoring nurses' activities, elderly care activities, and human health. Activity recognition systems require sufficient labeled training data and the shortage of this training data degrades their performance. This problem of insufficient data is solved by using the TransAct model which helps recognize activities in a new environment by transferring

[2] This effect is more visible with piano



(a) Guitar in $\frac{3}{4}$ time

(b) Guitar in $\frac{4}{4}$ time

(c) Piano in $\frac{3}{4}$ time

(d) Piano in $\frac{4}{4}$ time

Figure 3: Timing vs. timbre in spectrograms

knowledge from source data in a different environment [15]. This model can identify activities with limited training data in the target environment.

Another example is dermal image classification, where the three-class skin lesion classification problem is investigated using transfer learning. In this approach, the authors modified the pre-trained AlexNet model [16] by replacing the last layer with their own layers to make it compatible with their three-class problem. They also added two dropout layers to avoid overfitting. [10].

One more example is sentiment classification, where the task is to classify product reviews automatically, e.g. into positive and negative perspectives for a brand of laptop. For this task, first the reviews on items must be gathered and then a classifier is trained on the reviews with their corresponding labels. To obtain meaningful results, a large amount of labeled data is needed because the wording of reviews for various types of items can be very different. However, it is very expensive to do data-labeling. To reduce this effort, transfer learning can be used by adapting a classification model that is trained on some items to help learn classification models for some other item [3].

Transfer learning has been successfully applied to many other applications, such as heterogeneous transfer learning for image classification by exploring knowledge transfer from auxiliary unlabeled images and text data [28], Latin and Chinese character recognition [7], detecting a user's device location based on previously collected WiFi signals [22], lung pattern analysis [6], visual categorization [19], and so on. More detail on transfer learning applications can be found in a survey by Sinno Jialin Pan and Qiang Yang [21].

There are also many examples where transfer learning is used in Music Classification. Hamel et al. [13] proposed an approach for genre classification using feature representation transfer learning. They considered 4 distinctive datasets, each containing between 1000 to 3180 sound tracks, from 10 seconds in length to full song. To transfer knowledge between tasks, they learned a latent representation that was shared across tasks. This representation is learned by using an embedding method which consists of embedding both the features and the labels through transformation in a common space. The experiments are performed by extracting mel-spectrum features from audio. Their experiments showed that by transferring a representation between tasks, the classification accuracy can meaningfully improve when the number of training examples is inadequate. Our approach is different because the transfer is between more distant tasks, i.e. real world image classification to music classification.

Lee et al. [17] proposed a convolutional neural network (CNN) architecture for music auto-tagging with multi-level and multi-scaled features, i.e. with tags that are highly varied and have different levels of abstraction. Their approach is similar to image classification by taking 2D audio spectrograms as input data. This architecture is trained in three steps. First, the local audio features are captured using a set of CNNs, trained in supervised manner with the tag labels, taking different input sizes. Second, the audio features are extracted from all layers of the pre-trained CNNs and aggregated into a single feature vector. Lastly, the final prediction of tags is performed from the aggregated feature vector by putting them in a fully-connected neural network. This architecture can perform transfer learning by conducting the first step with one large dataset and the last two steps with another dataset. It also features a smaller distance between the sourse and the target task than our approach.

Another example of transfer leaning in music classification is an approach in which the authors, Choi et al., used a convolutional neural network which has already been trained for a source task, in this case music tagging. Input to the network is mel-spectrograms of the audio to be classified. They extract features from all convolutional layers and these features are concatenated to form a single vector called the convnet feature. This feature is then used to train

another network [5] . Our approach is simpler because we are using an already trained CNN and replacing the last layer with a new layer to make it compatible with our domain.

Finally, Van den Oord et al. [27] considered the problem of transfer learning by supervised pre-training for audio-based music classification as well. First, they learned low-level features from audio spectrograms to train a supervised model for a source task, using the K-means algorithm. Next, these trained models were used to extract higher level features from other datasets that were used to train the target task.

Other work on transfer learning in music information retrieval include content-aware collaborative music recommendation systems in which a neural network was trained on semantic tagging information as a content model and was used as a prior in a collaborative filtering model [18], and convolutional recurrent neural networks for music classification [4].

4 EXPERIMENTAL SETUP

In order to train, validate, and test a classifier, one needs a dataset with sufficient number of labeled data for each of the classes. For this paper, we use several such datasets in our experiments. These include datasets that we created specifically for this research, as well as datasets that have been prepared by other researchers and have been presented in other publications. The datasets we created contain approximately 600 audio clips in each class, a number that has been shown to be sufficient in transfer learning tasks [9, 25]. Our datasets are:

(1) **Genre** dataset: This dataset contains approximately 600 full-length songs in each of the following classes: Classical, Rock, and Metal. The genres were chosen so as to contain a pair of genres (Classical and Metal) that should be easily distinguishable by a human expert, as well as a pair of genres (Rock and Metal) where overlap was a lot more likely. Genre classification is one of the most common music classification tasks, which is why our first dataset addressed this problem. We also created a version of this dataset that contains only 30 seconds from each song (to avoid misrepresenting the song due to long fade-ins we chose seconds 30 to 60). We refer to this version as the **Short Genre** dataset.
(2) **Instrument** dataset: This dataset contains approximately 600 audio clips in each of the following classes: Acoustic Guitar, Classical Guitar. The two instruments were chosen because they are close in nature, but still distinguishable due to the different timbre of steel strings (acoustic guitar) to that of nylon strings (classical guitar). Each clip is 30 seconds long and the guitar is the only instrument, i.e. there are no vocals or other instruments.
This classification task is related to genre classification as differentiation will be based on timbral patterns. The selection of the two instruments was meant to provide a challenge for our classifier.
(3) **Vocalist** dataset: This dataset contains approximately 600 audio clips in each of the following classes: Female, Male. Each 30-second clip is part of a song with full instrumentation that features a vocalist singing either unaccompanied or with background vocals. The clips were created so that they feature singing throughout, and they cover a wide variety of genres (pop, rock, country, folk, rap etc.)
This dataset was created as an even harder challenge for our classifier. While female and male vocals differ in timbre, the remaining instruments would superimpose timbral patterns that would be common between the two classes. We wanted to see how well our classifier can handle this situation.
(4) **Composer** dataset: This dataset contains approximately 600 audio clips in each of the following classes: Beethoven, Mozart. The two composers were chosen due to their prolific nature, as well as the fact that they are likely to use similar instrumentation (Beethoven was born only 14 years after Mozart). The clips span a wide variety in terms of recording quality, instrumentation, and musical form, i.e. concerts, sonatas, opera etc. Each clip is 30 seconds long.
This dataset was meant to be the hardest challenge for our classifier. Only expert human classifiers would be able to perform well in such a task, and the distinction would not be based on timbral patterns but on higher level audio features, such as musical motifs, global structure etc. We expected our classifier to perform poorly in this task.

The existing datasets we used are:

(1) Extended Ballroom dataset [20]: This dataset contains audio clips in each of the following classes: Chacha, Foxtrot, Jive, Pasodoble, Quickstep, Rumba, Salsa, Samba, Slowwaltz, Tango, Viennesewaltz, Waltz, Wcswing. Each clip is 30 seconds long. For 9 of these classes, the cardinality of each class varies from 252 to 529, while the remaining 4 are smaller (cardinality varies from 23 to 65). For this reason, we also experimented with subsets of this dataset that contain its largest classes. We refer to the full dataset as **Ballroom**, while subsets contain the number of classes as a suffix, e.g. **Ballroom9** contains the 9 largest classes only.
(2) NSynth [11]: This is a large dataset that contains 305,979 audio clips. Each clip is 4 seconds long and it contains a single note. The notes contained include all notes in western music played in a large variety of acoustic, electronic, or synthetic instruments at 5 different velocities per note. We used parts of this dataset to create the following datasets for our experiments:
   (a) **NSynth Family**: This dataset contains 880 audio clips in each of the following classes: Acoustic, Electronic, Synthetic. Each class contains the full range of 88 notes that can be produced on a piano played at 5 different velocities on various acoustic, electronic, or synthetic keyboards.
   (b) **NSynth Instrument**: This dataset contains at least 650 audio clips in each of the following classes: Brass, Flute, Guitar, Keyboard, Mallet, Reed, String, Vocal. All clips in the dataset are from the acoustic family of instruments. There are small differences in the cardinalities of the classes as some instruments cannot produce the full range of western pitches.
   (c) **NSynth Organ**: This dataset contains 440 audio clips in each of the following classes: Electronic Organ 1, Electronic Organ 2. The only difference between the clips in the two classes is that they are produced by a different

Table 1: Numerical details about the datasets used in this paper

Dataset	Number of classes	Sorted Cardinalities	Clip length
Genre	3	600,600,600	Variable: Full songs
Short Genre	3	600,600,600	30 sec
Instrument	2	600,600	30 sec
Vocalist	2	600,600	30 sec
Composer	2	600,600	30 sec
Ballroom	13	529,507,497,470,468,464,455,350,252,65,53,47,23	30 sec
NSynth Family	3	880,880,880	4 sec
NSynth Instrument	8	880,880,880,760,730,690,670,650	4 sec
NSynth Organ	2	440,440	4 sec

instantiation of the same instrument, an electronic organ. This should correspond to the smallest differences in timbre present in the NSynth dataset.

Table 1 presents a concise description of all datasets used in this paper.

For every dataset described above, we experimented with two different versions:

(1) **Unprocessed**. In this version, the clips are as originally recorded.
(2) **Normalized**. In this version, each clip has been normalized so that the maximum peak loudness is 0dB FS. The reason for the normalization is so that the classifier cannot utilize loudness for classification. For example, classical music clips are typically less loud than metal ones. The normalization process makes all clips have the same maximum loudness (average loudness may still differ depending on the dynamic range of the clip).

In the following, we refer to each dataset as described above, e.g. the Unprocessed Short Genre dataset, or the Normalized Instrument dataset.

To begin each experiment, all music files are converted to spectrograms using the asperes tool [1]. The conversion process uses a logarithmic frequency scale with 12 bands per octave, which is appropriate for musical signals. The spectrograms created by asperes are black and white. We experimented with color spectrograms created by sox [12], where the intensity of the magnitude of each frequency band is represented by a distinct colour rather than a greyscale value, but there was no significant effect in the results presented in the next section.

Once all music clips have been converted to spectrograms, the music classification problem is transformed to an image classification problem. We can then apply existing image classifiers. For this paper, we used the Inception-v3 classifier [24], as it is freely available and open source. Inception-v3 is trained for the ImageNet Large Visual Recognition Challenge using the data from 2012. This is a standard task in computer vision, where models try to classify entire images into 1000 classes, like "Zebra", "Dalmatian", and "Dishwasher" [14]. We removed the last layer and replaced it with a fully-connected layer that reduces its input vector that contained 2048 features to an output vector whose size matched the number of classes in each experiment.

The same hyper-parameters were used across all experiments to avoid introducing bias. All parameters were set at their default value, with the exception of increasing the number of training iterations to 8000.

In the next section, we present the results we obtained with our approach on all the datasets described above. These results led us to conduct further experiments with modified versions of our datasets. We present these modified datasets and their results in Section 6.

5 EXPERIMENT RESULTS

Table 2 presents all the results we obtained from the Inception v3 classifier for the datasets presented in the previous section. It also presents accuracy results from a *baseline classifier*. This classifier uses input produced by Marsyas [26], a well-known audio feature extraction tool that extracts relevant features directly from the audio signal. These features were used as input to a Random Forest classifier that performed the classification. As it is based on a special-purpose system for audio classification, we expected that the baseline classifier would be more accurate than our approach, and act as an indication of the ceiling of a black box transfer learning approach for music classification.

Before discussing these results, it is important to note that, due to randomization, subsequent runs of the same experimental setup may produce slightly different results. These differences were never more than two percentage points. In Table 2, we report the average accuracy value over 5 runs. We also do not consider differences in accuracy that are smaller than 2% to be significant.

The same results are presented in graph form in Figures 4 and 5.

Based on these results, there are a number of observations that can be made:

(1) Overall, the results clearly indicate that a black box transfer learning approach, such as the one employed in this paper, can be very effective in a variety of music classification tasks. We discuss individual experiments below, but the achieved accuracy led us to design a further experiment to stress the limits of our approach, as even the accuracy for the Composer dataset was higher than we expected. We present this experiment in the next section.
With regard to our baseline, it was very encouraging to see that our approach performed at the same level in many

Table 2: Accuracy results for the Inception v3 and the baseline classifier

Dataset	Number of Classes	Inception v3 (Unprocessed)	Inception v3 (Normalized)	Baseline (Unprocessed)	Baseline (Normalized)
Genre	3	87.2	85.8	97.1	97.0
Short Genre	3	86.8	87.4	85.8	85.6
Instrument	2	96.3	89.4	97.3	97.4
Vocalist	2	78.2	77.6	96.9	96.7
Composer	2	64.5	73.9	80.9	81.9
Ballroom	13	54.8	62.3	54.8	54.7
NSynth Family	3	96.6	100.0	100.0	99.6
NSynth Instrument	8	97.3	97.1	99.7	100.0
NSynth Organ	2	100.0	100.0	100.0	100.0

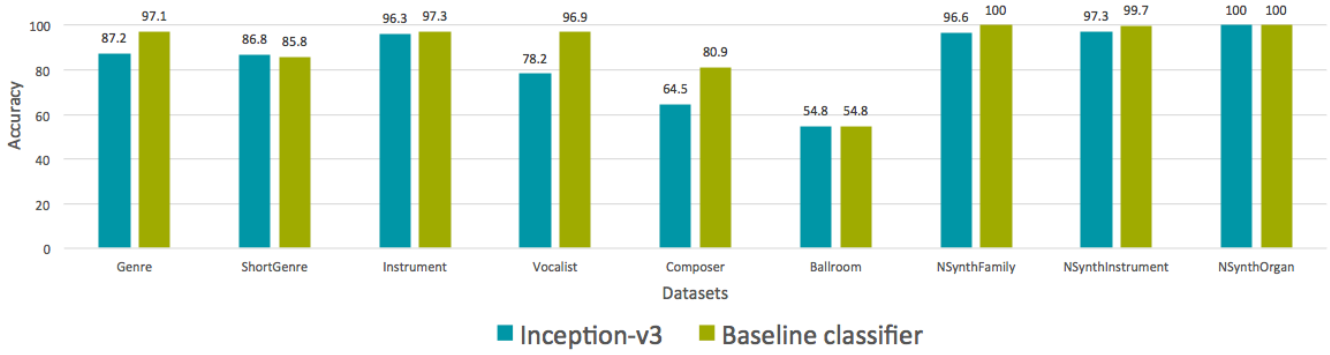


Figure 4: Accuracy results for Unprocessed Datasets

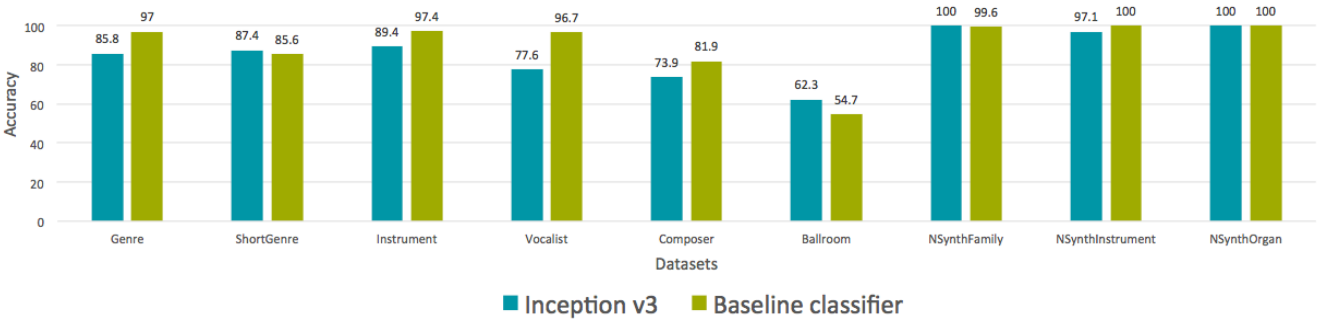


Figure 5: Accuracy results for Normalized Datasets

experiments, and even outperformed a special-purpose approach in the case of the Normalized Ballroom dataset. To be sure, the baseline approach performed clearly better in some experiments, such as with the Genre and Vocalist datasets.

(2) The Genre dataset was our pilot dataset. We used it to gauge the feasibility of our approach, as well as to determine how important of a factor is the length of the audio input.
When classifying Metal and Classical audio clips only, the accuracy of our approach was 99.1%. Even when another genre, such as Rock, was added to the classification task, the accuracy remained high as shown on the table. This indicated to us that our approach has merit leading to the rest of the experiments.
The results for the Short Genre dataset, when compared to that for the Genre dataset, demonstrate that the accuracy of our classification is not impacted when the length of the audio input is reduced to 30 seconds as opposed to a whole song. This allowed us to perform the rest of the experiments using audio clips that are 30 seconds long, which sped up the process of converting audio to spectrogram significantly.
It is very interesting to note that the performance of the baseline classifier dropped significantly when the length of

the input audio was reduced to 30 seconds. This indicates a level of robustness with our approach that does not exist in the baseline classifier, something we plan to investigate further in the future.

(3) Normalizing the audio clips in the datasets to equal loudness has yielded a variety of results. To start with, normalization had no noticeable effect for the baseline classifier. This is probably due to a normalization internal stage that takes place prior to feature extraction.
When it comes to our classifier, in some cases, such as the Genre or Vocalist dataset, normalization makes no significant difference. For the Instrument dataset, normalization has degraded the accuracy of our approach. However, for the Composer and Ballroom datasets, normalizing improves the results.
A possible explanation for this phenomenon is the following: If the various classes in a dataset differ in loudness, the neural network may use this information to its advantage. For example, if the acoustic guitar clips in the Instrument dataset are consistently louder than the classical guitar ones, the network may use this confounding factor to classify rather than the timbre of the instrument. Normalizing for loudness would remove this extra help and lower the accuracy.
On the other hand, normalization to 0dBFS involves raising the overall loudness of an audio clip. This results in the corresponding spectrogram being much brighter, as if the exposure of the image was increased (see Figure 7 in the next section for an example). This makes the timbral patterns in the spectrogram more pronounced, i.e. more "visible" to the classifier. This should improve accuracy (assuming there was no confounding as described above in the unprocessed audio clips).
To determine whether the above hypothesis holds, we conducted a loudness study for our datasets. We present the outcome of that study in the next section.

(4) The Ballroom dataset yielded the results with the lowest accuracy. There are several possible reasons for this. First, it is the dataset with the largest number of classes (13). Second, some of these classes contain a very small number of clips (less than 70 for four of the classes). It is practically impossible for a neural network to learn with such a small sample number. Finally, what distinguishes these classes is the timing of the music, rather then the instrumentation which is often similar. To investigate these factors, we conducted further experiments with this dataset (presented in the next section).

(5) The results for the NSynth dataset confirm our hypothesis that a neural network that has been trained with real world images will be able to distinguish the timbre of different instruments. Each clip in the NSynth dataset contains a single note from one instrument. The music classifier has no problem classifying these correctly as they contain easy to identify patterns (see Figure 6).

Before moving on to the follow-up experiments, it is important to note that each of the experiments presented in this section required only a few minutes to propagate the spectrogram input through the pre-trained neural network, and an hour or two (depending on the size of the dataset) for the last layer classification. The CPU used was a dual Intel X5660 processor (no GPU). This is significantly faster than training a neural network from scratch which would require significantly more processing time on specialized hardware, such as GPUs.

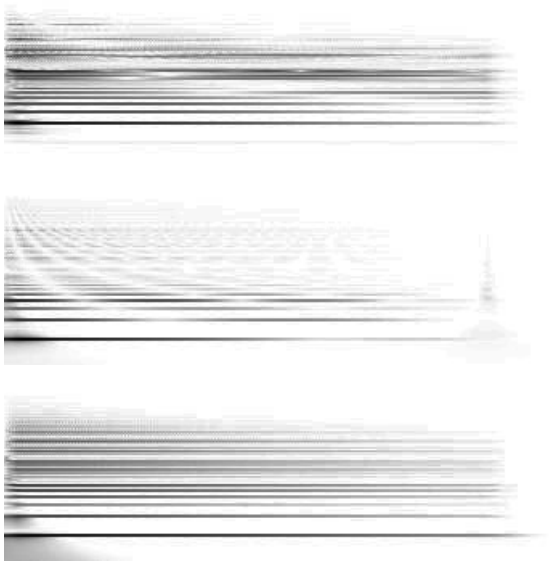


Figure 6: Spectrograms of an acoustic, electronic, and synthetic keyboard respectively playing the same note at the same velocity (from the NSynth dataset)

6 FOLLOW-UP EXPERIMENTS

Our first follow-up experiment concentrated on investigating the limits of our black box transfer learning approach by preparing a classification task that it would be ill-prepared to handle. Since a network trained on real world images would likely be insensitive to small changes in brightness, we rearranged the Ballroom dataset into 5 new classes with regard to the RMS loudness of each clip. For example, class Loud1 contained the one fifth of the audio clips that were the quietest, while class Loud5 contained the loudest clips etc. (loudness in the audio domain corresponds to brightness in the spectrogram domain).

The classification results confirmed our expectation that this would be a hard task for our classifier. We obtained an accuracy of 35.2%, by far our worse result. It is important to note however, that human error would probably be quite high for this dataset as well, as the boundary between classes is rather arbitrary. Accordingly, when we restrict the classification task to only two classes, Loud1 and Loud5, our accuracy increases to 79.5%. This means that our worse binary classification result remains the Composer dataset, which was the dataset we originally created to challenge our approach (73.9%).

Our next follow-up experiment involved the Ballroom dataset, for which we had the poorest results. We started by determining whether the fact that some of the classes had low cardinality was

Table 3: Accuracy results for Normalized Ballroom subsets

Dataset	Cardinalities	Number of classes	Accuracy
Ballroom9	>100	9	63.5
Ballroom8	>300	8	64.8
Ballroom7	>400	7	72.7
Ballroom2	>500	2	96.0

responsible for the low accuracy. Table 3 presents accuracy results with subsets of the Normalized Ballroom dataset that were created by removing classes with low cardinality. The choice of subset was driven by how closely cardinalities were clustered, i.e. we did not want to arbitrarily remove classes with similar cardinality to classes that would remain.

The results show some improvement but in most cases this can be attributed to the fact that there are fewer classes, i.e. less chance of error. Interestingly, when the dataset is reduced to two classes (Foxtrot and Waltz), we get the highest accuracy result for a normalized dataset (with the exception of NSynth). We believe this is due to distinct timbral elements in the two classes. Investigating the significance of this result is part of our future work.

We also conducted a further experiment with the Ballroom9 dataset to investigate whether data augmentation can improve our results. For this purpose, we created 8 copies of each audio clip in the dataset and normalized each clip to a different level at 5dB intervals. The quieter copy was normalized at -35dBFS (barely audible) and the loudest at 0dBFS (loudest possible without clipping). We then concatenated these 8 copies from quietest to loudest and created spectrograms for the augmented clips. Figure 7 presents an example.

When classifying with the augmented spectrograms as input to our black box transfer learning approach, the obtained accuracy was 68.1%, 4.6% higher than the non-augmented normalized version. It is possible that the clip repetition (even at different levels of intensity) is helpful to the neural network. A thorough study of how augmentation affects accuracy is part of our future work.

Our last follow-up experiment pertains to the effect of normalization to the accuracy of our approach. For this purpose, we computed the RMS loudness for every audio clip in our datasets and aggregated the results for every class. Table 4 shows the loudness data for most of the classes in our datasets (we omitted the 4 smallest classes in Ballroom, as well as the NSynth datasets where accuracy was already very high).

The data in Table 4 support the hypothesis outlined in the previous section. In particular, the two datasets that exhibited the most improvement due to normalization (Composer and Ballroom) are also the two datasets where loudness variations between classes are the smallest. In that case, normalization would not introduce bias, since all files will be affected similarly. This indicates that the improvement in accuracy is due to the timbral patterns in the spectrogram becoming more "visible" to the classifier.

On the other hand, the only dataset that showed a decline in accuracy due to normalization (Instrument), is also an outlier in terms of loudness disparity between its two classes (we have to go past one standard deviation in both distributions to have overlap). This indicates that the unprocessed dataset is closer to Acoustic vs Silence rather than Acoustic vs Spanish. After normalization, we get an accuracy measurement more in line with the other experiments, since the confounding associated with quietness is removed.

Table 4: RMS Loudness values for several of the classes in our datasets

Dataset : Class	**RMS Loudness (Average)**	**RMS Loudness (Std Deviation)**
Genre : Classical	0.06	0.03
Genre : Metal	0.21	0.08
Genre : Rock	0.16	0.06
Short Genre : Classical	0.06	0.03
Short Genre : Metal	0.20	0.09
Short Genre : Rock	0.15	0.07
Instrument: Acoustic	0.09	0.06
Instrument : Spanish	0.02	0.01
Composer : Beethoven	0.06	0.03
Composer : Mozart	0.06	0.03
Vocalist : Female	0.14	0.07
Vocalist : Male	0.18	0.10
Ballroom : Chacha	0.21	0.04
Ballroom : Foxtrot	0.17	0.05
Ballroom : Jive	0.22	0.05
Ballroom : Quickstep	0.19	0.05
Ballroom : Rumba	0.19	0.05
Ballroom : Samba	0.21	0.05
Ballroom : Tango	0.18	0.06
Ballroom : Vien. Waltz	0.19	0.06
Ballroom : Waltz	0.15	0.05

Finally, datasets that were not significantly affected by the normalization process, such as Genre and Vocalist, were somewhere in the middle in terms of loudness distribution and exhibited high values of standard deviation. This would seem to indicate that both phenomena outlined above occurred at the same time and counterbalanced each other. In other words, accuracy increased because the timbral patterns became more apparent, and at the same time decreased because classification was not aided by similarity to silence.

This last experiment shows clearly that normalization prior to classification of audio clips is necessary. Its benefits are twofold:

(1) Accuracy is increased because the timbral patterns in the spectrogram become more "visible" when loudness is raised.
(2) Unfair bias is removed for classes that are significantly quieter than others.

7 CONCLUSION

This paper presented a black box transfer learning approach to music classification by transferring knowledge gained by classifying real world images. The main contributions of the paper are:

- We showed that our approach is feasible, efficient, and accurate. By utilizing the minimum amount of domain knowledge possible, it can provide results that often equal those of special-purpose approaches.
- We showed that spectrograms are an appropriate representation for classification tasks that rely on timbral patterns.

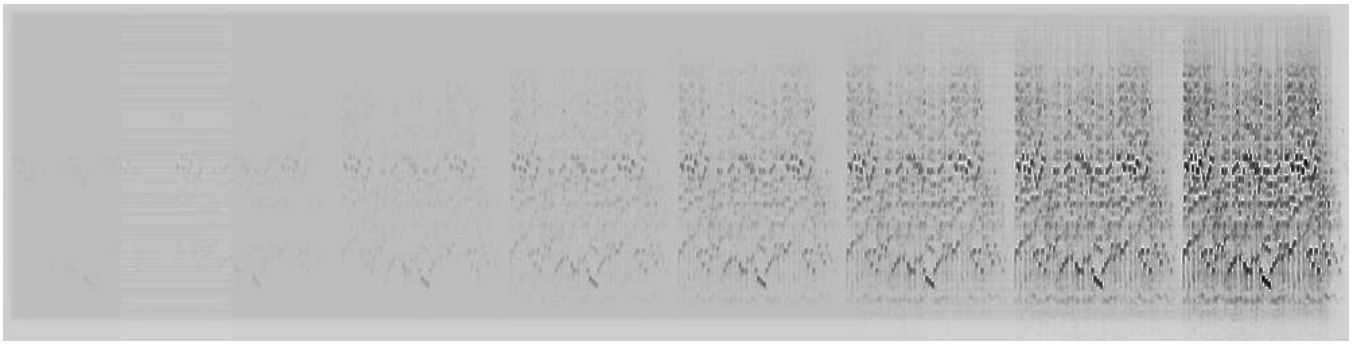


Figure 7: A spectrogram of 8 concatenated copies of an audio clip at different normalization levels. Note that the first few copies are barely visible

- We showed that our approach is more robust with regard to the length of a music clip, as our accuracy is not affected when it is reduced to 30 seconds (as opposed to the baseline classifier).
- We demonstrated the importance of normalizing music clips to 0dB FS to increase accuracy and remove confounding factors.
- We showed that data augmentation can be an important ally in music classification tasks.

There are several avenues for further research in this domain:

- Try different image representations for audio, such as waveforms, or spectrograms that contain phase information encoded as colour. Even though humans are insensitive to phase, our neural network has been trained with colour images. It would be interesting to see how it responds.
- Use a different pre-trained neural network, either one with a different and/or larger architecture, or one trained with a larger image dataset.
- Investigate data augmentation further. There are many possibilities on how to preprocess the spectrograms that could affect our accuracy results. We intend to experiment in this regard.
- Use our black box approach as a building block for a more elaborate classifier architecture, e.g. something similar to Sequential Minimal Optimization (SMO) [23], where each binary classifier is our black box.
- Try a different image classifier, such as the IBM Watson Visual Recognition service [8]. We were only able to use this service for this work in a limited fashion, due to the limits of a free account. However, preliminary results were very encouraging, especially when it comes to efficiency, so we intend to investigate further in the future.
- Experiment with different flavours of transfer learning, such as adding multiple layers to replace the original network's last layer, or re-training the whole network in the new domain.

REFERENCES

[1] Asperes - sound to image (spectrogram) and image to sound converter. 2004. (2004). http://www.findbestopensource.com/product/asperes

[2] Looking at Sound. 2005. (2005). http://betarhythm.blogspot.com/2005/11/looking-at-sound.html

[3] John Blitzer, Mark Dredze, Fernando Pereira, and others. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, Vol. 7. 440–447. http://anthology.aclweb.org/P/P07/P07-1.pdf#page=478

[4] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. 2017. Convolutional recurrent neural networks for music classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2392–2396. http://ieeexplore.ieee.org/abstract/document/7952585/

[5] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. 2017. Transfer learning for music classification and regression tasks. *arXiv preprint arXiv:1703.09179* (2017). https://arxiv.org/abs/1703.09179

[6] Stergios Christodoulidis, Marios Anthimopoulos, Lukas Ebner, Andreas Christe, and Stavroula Mougiakakou. 2017. Multisource Transfer Learning With Convolutional Neural Networks for Lung Pattern Analysis. *IEEE Journal of Biomedical and Health Informatics* 21, 1 (Jan. 2017), 76–84. https://doi.org/10.1109/JBHI.2016.2636929

[7] Dan C. Cireşan, Ueli Meier, and Jürgen Schmidhuber. 2012. Transfer learning for Latin and Chinese characters with deep neural networks. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*. IEEE, 1–6. http://ieeexplore.ieee.org/abstract/document/6252544/

[8] IBM Watson Developer Cloud. 2017. Visual Recognition | IBM. (2017). https://www.ibm.com/watson/developercloud/visual-recognition.html

[9] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition.. In *Icml*, Vol. 32. 647–655. http://www.jmlr.org/proceedings/papers/v32/donahue14.pdf

[10] Mohamed S. Elmahdy, Sara S. Abdeldayem, and Inas A. Yassine. 2017. Low quality dermal image classification using transfer learning. In *Biomedical & Health Informatics (BHI), 2017 IEEE EMBS International Conference on*. IEEE, 373–376. http://ieeexplore.ieee.org/abstract/document/7897283/

[11] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. 2017. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. *CoRR* abs/1704.01279 (2017). http://arxiv.org/abs/1704.01279

[12] SoX Sound eXchange. 2015. (2015). http://sox.sourceforge.net/

[13] Philippe Hamel, Matthew EP Davies, Kazuyoshi Yoshii, and Masataka Goto. 2013. Transfer Learning In MIR: Sharing Learned Latent Representations For Music Audio Classification And Similarity. In *ISMIR*. 9–14. https://staff.aist.go.jp/m.goto/PAPER/ISMIR2013hamel.pdf

[14] Inception-v3. 2016. (2016). https://www.tensorflow.org/tutorials/image_recognition

[15] Md Abdullah Al Hafiz Khan and Nirmalya Roy. 2017. TransAct: Transfer learning enabled activity recognition. In *Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on*. IEEE, 545–550. http://ieeexplore.ieee.org/abstract/document/7917621/

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

[17] Jongpil Lee and Juhan Nam. 2017. Multi-Level and Multi-Scale Feature Aggregation Using Pre-trained Convolutional Neural Networks for Music Auto-tagging. *arXiv preprint arXiv:1703.01793* (2017). https://arxiv.org/abs/1703.01793

[18] Dawen Liang, Minshu Zhan, and Daniel PW Ellis. 2015. Content-Aware Collaborative Music Recommendation Using Pre-trained Neural Networks.. In *ISMIR*. 295–301. http://dawenl.github.io/publications/LiangZE15-ccm.pdf

[19] Ling Shao, Fan Zhu, and Xuelong Li. 2015. Transfer Learning for Visual Categorization: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* 26, 5 (May 2015), 1019–1034. https://doi.org/10.1109/TNNLS.2014.2330900

[20] Ugo Marchand and Geoffroy Peeters. 2016. Scale and shift invariant time/frequency representation using auditory statistics: Application to rhythm description. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*. IEEE, 1–6. http://ieeexplore.ieee.org/abstract/document/7738904/

[21] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (Oct. 2010), 1345 – 1359. https://doi.org/10.1109/TKDE.2009.191

[22] Sinno Jialin Pan, Vincent Wenchen Zheng, Qiang Yang, and Derek Hao Hu. 2008. Transfer learning for wifi-based indoor localization. In *Association for the advancement of artificial intelligence (AAAI) workshop*. 6. https://vvvvw.aaai.org/Papers/Workshops/2008/WS-08-13/WS08-13-008.pdf

[23] John Platt. 1998. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. Technical Report. 21 pages.

[24] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2818–2826. http://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.html

[25] Tensorflow. 2016. How to Retrain Inception's Final Layer for New Categories. (2016). https://www.tensorflow.org/tutorials/image_retraining

[26] George Tzanetakis and Perry Cook. 1999. MARSYAS: A Framework for Audio Analysis. *Org. Sound* 4, 3 (Dec. 1999), 169–175. https://doi.org/10.1017/S1355771800003071

[27] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2014. Transfer learning by supervised pre-training for audio-based music classification. In *Conference of the International Society for Music Information Retrieval (ISMIR 2014)*. https://biblio.ugent.be/publication/5973853

[28] Yin Zhu, Yuqiang Chen, Zhongqi Lu, Sinno Jialin Pan, Gui-Rong Xue, Yong Yu, and Qiang Yang. 2011. Heterogeneous Transfer Learning for Image Classification.. In *AAAI*. https://www.researchgate.net/profile/Zhongqi_Lu/publication/221605039_Heterogeneous_Transfer_Learning_for_Image_Classification/links/0046352ca689c9f097000000.pdf