Anuvaadya:
Instrumental Music Translation

Ansh Khurana 170050035
Anuj Diwan 170070005
Soumya Chatterjee 170070010

1. Introduction

Music is often identified by genres, the instrument it's played on and the musical notes being played. Musical notes provide an evidence about the underlying structure of music and it's 'content'. Many musical compositions involve playing two or more instruments simultaneously in harmony. Thus it is natural to associate 'content' and a style dependent on the instrument used to play music. In this project we tackle the problem of converting music from one instrument to the other, while preserving the content of the music. This problem can find applications in professional music production software, audio mixing and music generation. The idea of preserving the 'content' of a sample by changing its 'style' is not new and has been thoroughly explored in the computer vision domain. After studying and comparing previous works, we focus on an Autoencoder approach to solve the music translation problem. Autoencoders are trained in an unsupervised fashion for reconstructing the perturbed input. We wish to apply the conversion task for recordings taken from Indian instruments like the Tabla or Mridangam and translate them to western instruments. This approach is apt for our project since supervised music recording with notes transcription and parallel data for multiple Indian instruments is extremely scarce. Taking inspiration from various approaches used in the WaveNet Autoencoder to improve upon the translation task, we conduct experiments on simpler RNN based networks. We build a smaller LSTM (4 hidden layers each for the encoder and decoder) based Autoencoder (which is more practical in scale, given the training resources) and study the gains of using techniques like Attention in the network architecture.

2. Prior Work

2.1. Domain Transfer

Recently, there has been a considerable amount of work, mostly on images and text, which performs unsupervised translation between domains $\mathcal{A}$ and $\mathcal{B}$ without being shown any matching pairs, i.e., in a completely unsupervised way. Almost all of this work employs GAN constraints in order to ensure a high level of indistinguishability between the translations of samples in A and samples from the domain B. In the WaveNet based Encoder-Decoder, the output is generated by an autoregressive model and training takes place using the ground truth output of the previous time steps ("teacher forcing"), instead of the predicted ones. A complete autoregressive inference is only done during test time, and it is not practical to apply such inference during training in order to get a realistic generated ("fake") sample for the purpose of training the GAN.

2.2. Audio Synthesis

Recent contributions in Text-To-Speech(TTS) have successfully conditioned WaveNet on linguistic and acoustic features to obtain state of the art performance. In our encoder-decoder architecture, we use WaveNet as the output of the decoder, and backpropagate through it down to the encoder. Voice conversion can be obtained by employing a variational autoencoder that produces a quantized latent space that is conditioned on the speaker identity. In the supervised learning domain, an audio style transfer between source and target spectrograms has been performed with sequence-to-sequence recurrent networks. This method requires matching pairs of samples played on different instruments. In another fully supervised work, a graphical model aimed at modeling polyphonic tones of Bach was trained on notes, capturing the specificity of Bach's chorales. This model is based on recurrent networks and requires a large corpus of notes of a particular instrument produced with a music editor.

2.3. Style Transfer

In the task of style transfer, the "content" remains the same between the input and the output, but the "style" is modified. Notable contributions in the field use methods synthesize a new image that minimizes the content loss with respect to the content-donor sample and the style loss with respect to one or more samples of a certain style. The content loss is based on comparing the activations of a network training for an image categorization task. The style loss

compares the statistics of the activations in various layers of the categorization layer. Attempts have been made at audio style transfer.

3. Methodology

3.1. WaveNet Autoencoder

This method is based on a multi-domain WaveNet autoencoder, with a shared encoder and a disentangled latent space that is trained end-to-end on waveforms. The encoder (E) is trained to be a universal extractor for the music content and provides domain-invariant representation of the input. Each domain has it's own decoder D_j which is trained only the recordings from domain j.

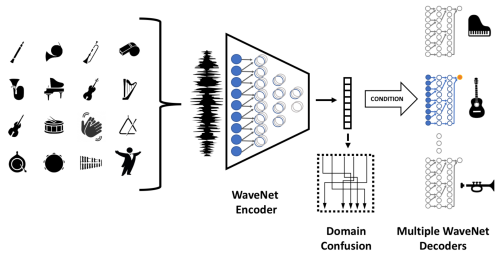

Figure 1. Overview of the Architecture

while the decoder is unique for each instrument. The encoder consist of 3 blocks of 10 residual layers each consisting of WaveNet like non-casual (depending on future inputs) dilated convolution with increasing kernel size. This is followed by a 1X1 convolution and average pooled to get the encoded input. The activation function used for each of these layers is ReLU. The Domain Confusion Network is a simple 1x1 convolutional network followed by global average pool and Softmax to give the probabilities of the input belonging to the training domains. The decoder for each instrument is a WaveNet decoder with input being the same as the original recording and the conditioning is given as the output of the encoder.

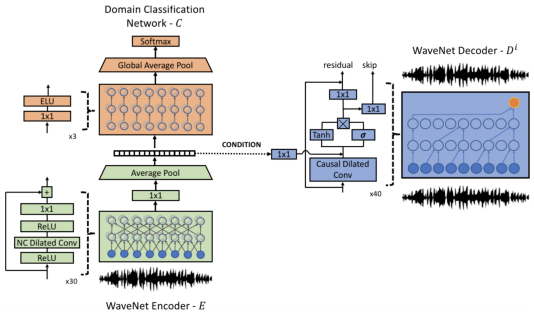

Figure 2. WaveNet Auto Encoder Architecture

3.1.1 Audio Input Augmentation

In order to improve the generalization capability of the encoder, as well as to enforce it to maintain higher-level information, we employ a dedicated augmentation procedure that changes the pitch locally. The resulting audio is of a similar quality but is slightly out off tune. This is enables the network to be more generalized for instruments that are unseen during training.

3.1.2 Domain Confusion Network

The key to being able to train a single encoder architecture is making sure that the domain-specific information is not encoded. We do this using a domain confusion network that provides an adversarial signal to the encoder. The Domain Confusion Network tries to classify the domain from which the input came from using output of the encoder. In this adversarial setting, the encoder tries to maximize classification uncertainity of the Domain Confusion Network.

3.1.3 Architecture

The network is trained end-to-end for each training example s. The universal encoder is shared across all domains

3.1.4 Training

During training, the network is trained as a denoising Autoencoder, which recovers the undistorted version of the original input. Since the distorted input is no longer in the musical domain of the output, the network learns to project out-of-domain inputs to the desired output domain. For a sample s^j from domain j, a random input augmentation (as described in section 3.1.1) is applied with a random seed r. Let the resultant input be $O(s^j, r)$. Let E be the universal encoder and D_j be the decoder for domain j. The Domain Confusion Network C is being trained to classify the encoder output $E(O(s^j, r))$ into the input domain by minimizing the Cross Entropy Loss ($\mathcal{L}$). The autoencoders $j = 1, 2..$ are trained with the loss

$$\sum_j \sum_{s^j} \mathop{\mathbb{E}}_r \mathcal{L}\left(D^j\left(E\left(O\left(s^j, r\right)\right)\right), s^j\right) - \lambda \mathcal{L}\left(C\left(E\left(O\left(s^j, r\right)\right)\right), j\right)$$

while the domain confusion network minmizes the classification loss

$$\sum_j \sum_{s^j} \mathop{\mathbb{E}}_r \mathcal{L}\left(C\left(E\left(O\left(s^j, r\right)\right)\right), j\right)$$

3.1.5 Inference

To transform an input sample s from any domain (even unseen domains), to output domain j, we pass the input through auto encoder j without applying the random distortion. Thus, the transformed sample is given by $D^j(E(s))$.

3.2. LSTM Encoder-Decoder

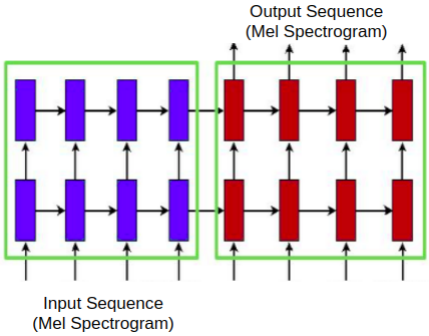

Figure 3. Architercture for LSTM Encoder-Decoder Architecture

3.2.1 Features used

We use Mel spectograms as the features for the audio files. 64 filterbanks are used. Input features are normalized before passing them to the encoder.

3.2.2 Architecture and Loss function

An LSTM based encoder-decoder architecture with a shared encoder E and multiple decoders D_j for each instrument in the training set. The shared encoder consists of a stacked unidirectional LSTM with 4 layers. The last hidden state of the encoder is provided as the initial input to the decoder. The decoder consists of a stacked unidirectional LSTM with 4 layers followed by a linear layer. Teacher forcing is used to decide between using the ground truth and the previous output for the next model input.
The mean squared error loss function is used with the Adam optimizer.

3.3. LSTM Encoder-Decoder with Attention

NOTE: Features are same as the previous.

3.3.1 Architecture

Encoder architecture is same as the previous. Now, the attention mechanism is added. The attention network takes all the encoder states H and the previous decoder state s_{t-1} as input. Then, the attention mechanism first computes an *energy* between these as follows:

$$E_t = \tanh\left(\text{attn}\left(s_{t-1}, H\right)\right)$$

where attn is a linear layer of size $(decoder\_hidden\_dim, num\_enc\_states)$. This computes how well each encoder hidden state 'matches' the decoder state. We linearly collapse this to a vector of length (num_enc_states) using v and apply a final softmax as follows:

$$a_t = \text{softmax}\left(vE_t\right)$$

This attention vector is then used to perform a weighted average of the encoder states H, giving w_t. Finally the decoder input is the vector w_t and the original input vector concatenated, and the previous decoder hidden state. The decoder architecture is same as previous and teacher forcing is used here too.

3.4. LSTM Encoder-Decoder with Attention and Domain Confusion

Most of the architecture is same as the previous. A new network, the domain confusion network is used. The last encoder output is given as input to the network. The network is a simple logistic regression that is trained using the cross entropy loss to try and predict the class of the source instrument. The Seq2Seq model is now trained to minimize the difference of the original loss and the domain confusion loss.

4. Implementation Details

4.1. WaveNet Auto Encoder

The model architecture has been defined and trained using PyTorch.

The following parameters are used while training the network: {batch-size : 2, learning rate : 0.001, lr_decay : 0.995, number of layers: 14, number of blocks : 4, input_augmentation : on}

We trained the network for music translation from any pair in the following domains : {Cello, Tabla, Violin, Piano, Quintet}. While training data for the western instruments was obtained from the MusicNet dataset, we requested for access to the Tabla dataset on Dunya. The wav files used for traininf are compressed to reduce the dynamic range of the audio, using sox command. This data is then read into a numpy array using 8-bit mu-law encoding using the scipy.io.wavfile module. It is then stored into h5 files on disk. To save training time, weights for encoder network were extracted from a pre-trained network while the decoder was trained independently

4.2. Features used for all LSTM-based models

We extract Mel Spectrogram features from the input wav files using Librosa sampling at 16000 Hz. The spectrograms are extracted such that there are 64 filterbanks and the maximum frequency is clipped at 8000 Hz. These filterbanks are normalized since the larger frequencies have very low

maginitude. We observe that the normalization leads to a significant improvement in performance.

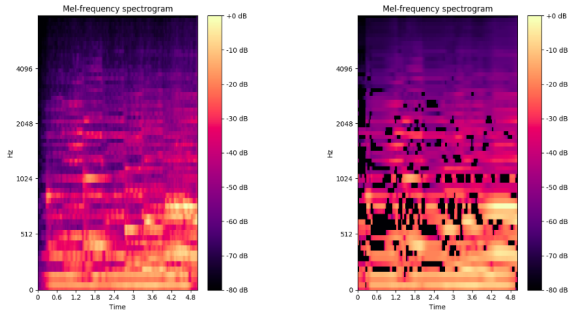

Figure 4. Original and Reconstructed Spectrograms

4.3. LSTM Encoder-Decoder

The encoder is a stacked LSTM with 4 layers. Each layer has 32 hidden nodes and the input is 64 dimensional. A dropout with probability of 0.2 is also used while training. The decoder architecture is exactly the same. While training the decoder, teacher forcing is used with a probability of 0.5. All the parameters are initialized uniformly within the range (-0.08, 0.08).

4.4. LSTM Encoder-Decoder with Attention

The Attention based model is built on top of the previous LSTM Encoder-Decoder model but the encoder is a bidirectional model. It shares the above hyperparameters. To apply Attention to this network, we add a fully connected layer to calculate the attention weights from the encoder and decoder hidden states, described in the architecture.

4.5. LSTM Encoder-Decoder with Attention and Domain Confusion

The architecture is the same as above except the addition of a Domain Confusion Network which is a single layer neural network with as many outputs as the number of instruments during training. It is trained using cross entropy loss.

5. Experiments, Results, and Discussion

5.1. Results of WaveNet Autoencoder

Use this Google Drive link to access the Google Drive folder containing all of the following samples:

Input Instrument	Original	Converted to Cello
Cello	Link	Link
Tabla	Link	Link
Violin	Link	Link
Piano	Link	Link
Quintet	Link	Link

Table 1. Results of translation to Cello

5.2. Comparison of Final losses

Architecture	Loss at Saturation
LSTM Encoder-Decoder	0.803456
LSTM Enc-Dec w/ Attn.	0.634245
LSTM Enc-Dec w/ Attn. and Domain Confusion	0.524566(Enc-Dec Loss) 0.664352(Domain conf.)

Table 2. Losses at saturation. All values are for mean squared error loss except Domain Confusion Loss which is cross entropy loss

6. Conclusion

A few drawbacks of the Autoencoder architecture include the large amount of diverse training data required for it to be robust against overfitting to the training instruments. Employing a diverse training dataset and large training capacity does imply that the network will be able to generalize to even unseen instruments, it makes it intractable for the network to be trained and evaluated on weaker GPUs. However, the Autoencoder approach is unsupervised and does not rely on supervision in the form of matched samples between domains or musical transcriptions like notes.

Our LSTM based approaches help identify the benefits of various different parts of the final LSTM with Attention and Domain Confusion network model. We observe a significant improvement in performance in going from a simple Encoder-Decoder LSTM to one with attention. Similarly, adding the domain confusion network results in a boost in performance.

The present WaveNet based Encoder-Decoder architecture, we demand too much from the shared encoder by depending on it learn a perfectly instrument independent subspace for mapping the inputs to. In future, we would like to relax this strict constraint by allowing the encoder to also have some instrument specific outputs (in addition to the shared ones), i.e. a private subspace for each instrument in the training data in addition to the shared subspace. This data will be passed to the instrument specific decoders during training. While testing on unseen instruments (by the encoder), this private subspace will not be used. This gives more flexibility to the encoder and is expected to improve performance.

References

[1] https://arxiv.org/abs/1805.07848, A Universal Music Translation Network

[2] https://github.com/facebookresearch/music-translation, Reference implementation from FAIR

[3] https://ytaigman.github.io/musicnet

[4] https://dunya.compmusic.upf.edu/