WaveNet Autoencoder with Contrastive Predictive Coding for Music Translation

Chester Huynh, Silu Men, David Shi, Maggie Wang

1 Introduction

Background In image-to-image translation, the goal is to transform an input image in the source domain to an output image that bears the appearance of the target domain but preserves the content of the input [1]. A similar idea can be applied to musical audio-to-audio translation. We develop an unsupervised music translation model capable of translating a performance of a piece (content) played by one instrument (appearance) into a performance of the same piece played by another instrument.

Related Work The Universal Music Translation (UMT) network [2] uses a WaveNet (WN) autoencoder to translate between instrument domains, where a single encoder is shared across domains and each has a separate decoder. A domain confusion discriminator on the latent space discourages domain-specific embeddings.

Contrastive Predictive Coding (CPC) [3] seeks to preserve the mutual information between the current context of the signal and future signal data points by making predictions in latent space. CPC uses an encoder, g_{enc}, to map the input x to latent representations z, which are then fed into an autoregressive model, g_{ar}, to produce context latent representations c. The model is trained to minimize InfoNCE, a probabilistic contrastive loss.

2 Methods

Dataset We performed training and inference on the MusicNet dataset [4], and evaluated on 3 domains: Solo Cello, Solo Violin, and Beethoven Solo Piano. We used an 80-10-10 split, yielding 32963 sec of training data, 4683 sec of validation data, and 5001 sec of testing data. We further evaluated our model on samples of 0.5-second, single-pitch, monophonic recordings of keyboard and strings from the NSynth test set [5].

Adding CPC to UMT In an attempt to better capture long-range temporal dependencies, we replaced UMT's encoder with CPC while preserving UMT's discriminator and WaveNet decoders. Fig. 3 provides a high-level view of our architecture. For CPC's g_{enc}, we used a 6-layer strided convolutional neural network. For CPC's g_{ar}, we tried two variants: a GRU with a 64-dimensional hidden state (Fig. 4) and UMT's original WaveNet-like encoder module (Fig. 5). We refer to the full model variants as CPC_{GRU}+WN_{dec} and CPC_{WN}+WN_{dec}.

z and c were used to compute the InfoNCE loss (Eqn. 1) with 5-step-ahead predictions (Algorithm 2). c was upsampled and used as the conditioning signal for each domain-specific WaveNet decoder.

Let x_j denote a raw audio sample from the j^{th} domain, c denote the output of g_{ar}, D_j denote the WaveNet decoder corresponding to the j^{th} domain, and C denote the domain confusion discriminator. The overall loss function of our model was

$$\sum_j \sum_{x_j} \mathcal{L}(D_j(c), x_j) - \lambda \mathcal{L}(C(x_j), j) + \text{InfoNCE}(z, c)$$

Training and Inference Due to limited compute, we performed training and inference on only the Solo Cello, Solo Violin, and Beethoven Solo Piano MusicNet domains. We prepended our g_{enc} and g_{ar} CPC modules to frozen pretrained UMT decoders.

CPC_{GRU}+WN_{dec} was trained for 100 epochs (25 min/epoch) on 0.625-second raw audio clips from the training sets of each of the domains using a batch size of 8 and an Adam optimizer with learning rate = 5e-4 and learning rate decay = 0.995 after each epoch. We used mainly UMT default hyperparameters, with a few modified for hardware compatibility. Due to time constraints, CPC_{WN}+WN_{dec} was trained for only 80 epochs. During inference, for each instrument domain, we sampled one 2-sec raw audio clip from the testing set and translated the clip to all three instrument domains.

Evaluation For both datasets, we qualitatively listened to translated samples. For MusicNet, we ran a cycle consistency test on a small subset ($n = 2$). For NSynth, we translated 0.5-sec single-pitch recordings of acoustic keyboard and strings to the cello, violin, and piano domains and measured the similarity between each translated recording and an original recording in the output domain using normalized cross-correlation (NCC) [5]. We also visually inspected chromagrams, a pitch class profile, to assess how well pitch was preserved after translation. A good translation would correlate well with the original and preserve pitch.

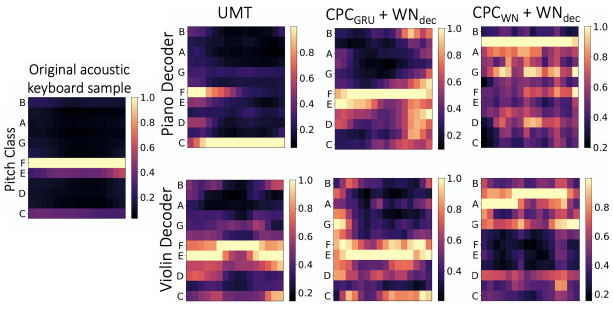

Figure 1: Chromagrams for acoustic keyboard sample translated to piano and violin using UMT, CPC_{GRU}+WN_{dec}, and CPC_{WN}+WN_{dec}.

3 Results

Qualitative The MusicNet translated samples (available here) generated using UMT had the least amount of perceivable noise, pitch distortion, and instrument blending, followed by CPC_{GRU}+WN_{dec} and CPC_{WN}+WN_{dec}. Chromagrams of the translated NSynth samples indicated that pitch preservation was comparable between CPC_{GRU}+WN_{dec} and UMT but that CPC_{WN}+WN_{dec} had relatively poor pitch preservation (Fig. 1).
Quantitative Table 1 shows our models have higher NCC on cello, comparable on violin, and worse on piano. Table 2 shows we perform better on cello, better on violin, and worse on piano.

Model	Cello	Violin	Piano
1	0.49 ± 0.20	0.55 ± 0.25	0.52 ± 0.26
2	$\mathbf{0.61 \pm 0.24}$	0.53 ± 0.25	0.41 ± 0.25
3	0.57 ± 0.28	0.53 ± 0.19	0.48 ± 0.24

Table 1: Average NCC between a translated and an original recording. Models: (1) pretrained UMT, (2) CPC_{GRU}+WN_{dec}, (3) CPC_{WN}+WN_{dec}.

4 Discussion

CPC_{GRU}+WN_{dec} and CPC_{WN}+WN_{dec} had overall worse performance than the baseline pretrained UMT model. This could be due to a variety of factors, including hardware-restricted hyperparameters and loss of semantic information from the input when downsampled by g_{enc}. CPC_{WN}+WN_{dec}'s poor performance could be attributed to our decision to compress the g_{enc} output into a single channel for compatibility with UMT's WaveNet encoder design. While this g_{ar} performed poorly, transformers, employed by the Jukebox model, [6] have had success in music generation. Though CPC_{GRU}+WN_{dec}'s quantitative NCC and chromagram results were comparable to UMT, a qualitative listen to the translated samples revealed worse quality. This could be due to our evaluation metric favoring pitch over other aspects of musicality, such as dynamics, tone, and tim-

bre. Though we did not achieve SOTA performance, $\text{CPC}_{\text{GRU}}+\text{WN}_{\text{dec}}$ is promising, so the idea of combining CPC with Wavenet decoders for musical audio-to-audio translation warrants further exploration.

References

[1] Taesung Park, Alexei A. Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive Learning for Unpaired Image-to-Image Translation. *arXiv:2007.15651 [cs]*, August 2020.

[2] Noam Mor, Lior Wolf, Adam Polyak, and Yaniv Taigman. A Universal Music Translation Network. *arXiv:1805.07848 [cs, stat]*, May 2018.

[3] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *arXiv:1807.03748 [cs, stat]*, January 2019.

[4] John Thickstun, Zaid Harchaoui, and Sham Kakade. Learning Features of Music from Scratch. *arXiv:1611.09827 [cs, stat]*, April 2017.

[5] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. *arXiv:1704.01279 [cs]*, April 2017.

[6] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A Generative Model for Music. *arXiv:2005.00341 [cs, eess, stat]*, April 2020.

[7] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. *arXiv:1609.03499 [cs]*, September 2016.

5 Appendix

5.1 InfoNCELoss Equation and Pseudocode

For an input sequence $x = \{x_1, \ldots, x_t\}$ with latent representation $z = \{z_1, \ldots, z_n\}$ and context latent representation $c = \{c_1, \ldots, c_n\}$, we compute the InfoNCE loss as

$$\mathcal{L}_{InfoNCE} = -\,\mathbb{E}_X \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \right] \tag{1}$$

where $f_k(x_{t+k}, c_t) = \exp(z_{t+k}^T W_k c_t)$ estimates the density ratio $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$, and $X = \{x_1, ..., x_N\}$ consists of one positive sample from $p(x_{t+k}|c_t)$ and $N-1$ negative samples from $p(x_{t+k})$.

Algorithm 1 Get the negative samples

procedure GET_NEG_Z($z, k, t,$ n_rep) ▷ The encoded sample, future time step, current time step and number of repetitions

  $\mathbf{Z} \leftarrow length(z)$ ▷ length of the encoded sample

  neg_idx $\leftarrow [0, 1, ..., \mathbf{t} + (k-1), \mathbf{t} + (k+1), ..., \mathbf{Z}]$ ▷ Make a range of neg_idx $\forall\ t$ in batch

  neg_samples $\leftarrow z[0, 1, ..., \mathbf{t} + (k-1), \mathbf{t} + (k+1), ..., \mathbf{Z}]$ ▷ Make the negative samples $\forall\ t$ in batch

  neg_samples $\leftarrow$ randomly sample with replacement rows in neg_samples

  return neg_samples ▷ Return the negative samples

end procedure

Algorithm 2 Forward pass of the InfoNCELoss

procedure FORWARD(z, c, n_rep) ▷ The encoded sample, context vector and number of repetitions

loss $\leftarrow 0$ ▷ initalize loss

$B \leftarrow length(z[0])$ ▷ get number of batches

$\mathbf{Z} \leftarrow length(z)$ ▷ length of the encoded sample

$t \leftarrow \text{randint}(0, \mathbf{Z})$ ▷ sample a random timestep t

$c\_t \leftarrow c[0 : B, t]$ ▷ get context vector for specific timestep

for $k \leftarrow 1, \text{prediction\_steps}$ **do**

$neg\_samp \leftarrow \text{GET\_NEG\_Z}(z, k, t, n\_rep)$ ▷ get negative samples

linear $\leftarrow Wk[k-1]$

pred $\leftarrow \text{linear}(c\_t)$ ▷ compute $W_k c_t$

pos_samp $\leftarrow z[\mathbf{t} + k]$ ▷ get positive sample at $t + k$ $\forall$ batches

$f\_k\_pos \leftarrow \text{diag}(\text{pos\_samp} * \text{pred})$ ▷ get positive matched batches' f_k's

$f\_k\_neg \leftarrow \text{diag}(\text{neg\_samp} * \text{pred})$ ▷ get negative matched batches' f_k's

$f\_k \leftarrow \text{stack}(f\_k\_pos, f\_k\_neg)$ ▷ stack positive and negative

$log\_f\_k \leftarrow mean(LogSoftmax(f\_k), dim = 2)$ ▷ apply logSoftmax and average across replicates

loss $\leftarrow$ loss $- log\_f\_k[:, 0]$ ▷ update loss with the positive sample $\forall$ batches

end for

loss $\leftarrow \frac{\text{loss}}{\text{prediction\_steps} \times B}$

return loss ▷ Return the calculated loss

end procedure

5.2 Supplemental Results

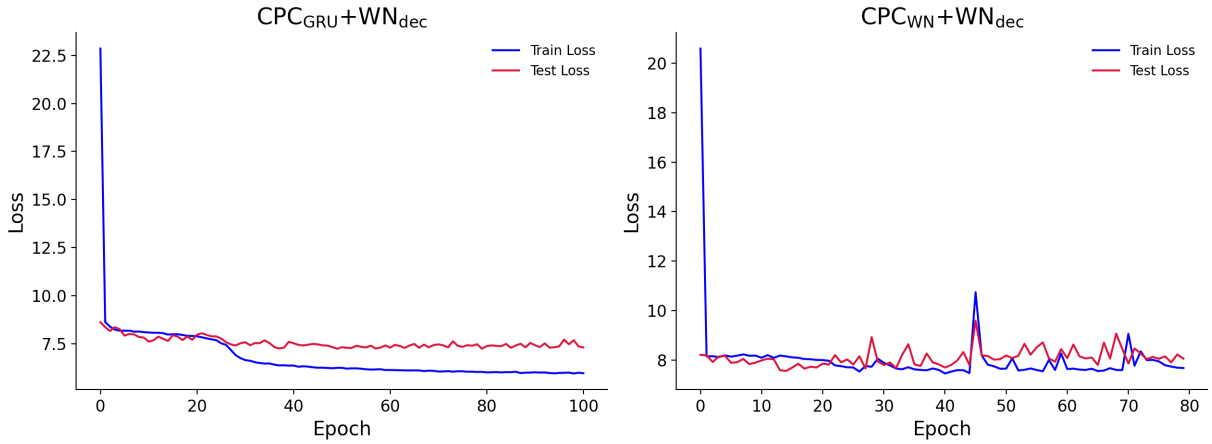

Figure 2: Average training and testing loss of CPC_{GRU}+WN_{dec} (*left*) and CPC_{WN}+WN_{dec} (*right*) models over 1000 mini-batches for each epoch. Due to time constraints, we trained CPC_{GRU}+WN_{dec} for 100 epochs and CPC_{WN}+WN_{dec} for 80 epochs. The loss for CPC_{WN}+WN_{dec} is evidently more volatile, which suggests that further modifications to the model hyperparameters could be made.

Model	Cello	Violin	Piano
1	0.92	0.40	0.84
2	**0.96**	**0.49**	0.58
3	0.80	**0.58**	0.61

Table 2: Average similarity scores on cycle consistency examples. Model 1: pretrained UMT, Model 2: CPC_{GRU}+WN_{dec}, Model 3: CPC_{WN}+WN_{dec}

5.3 Model Architecture

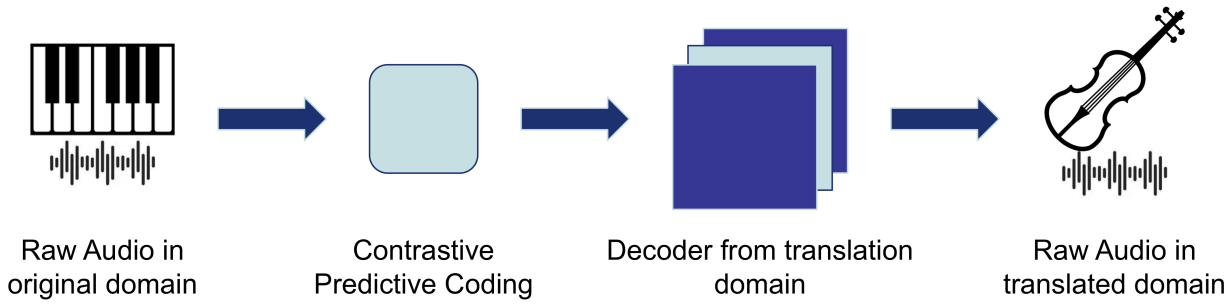

Figure 3: High-level architecture. We extend the encoding step of the UMT network [2] by applying contrastive predictive coding to produce context latent representations of the input audio sample rather than only a WaveNet encoder. We utilize the same decoder architecture and use the context latent sequences to condition the decoders and produce translated samples in another instrument domain.

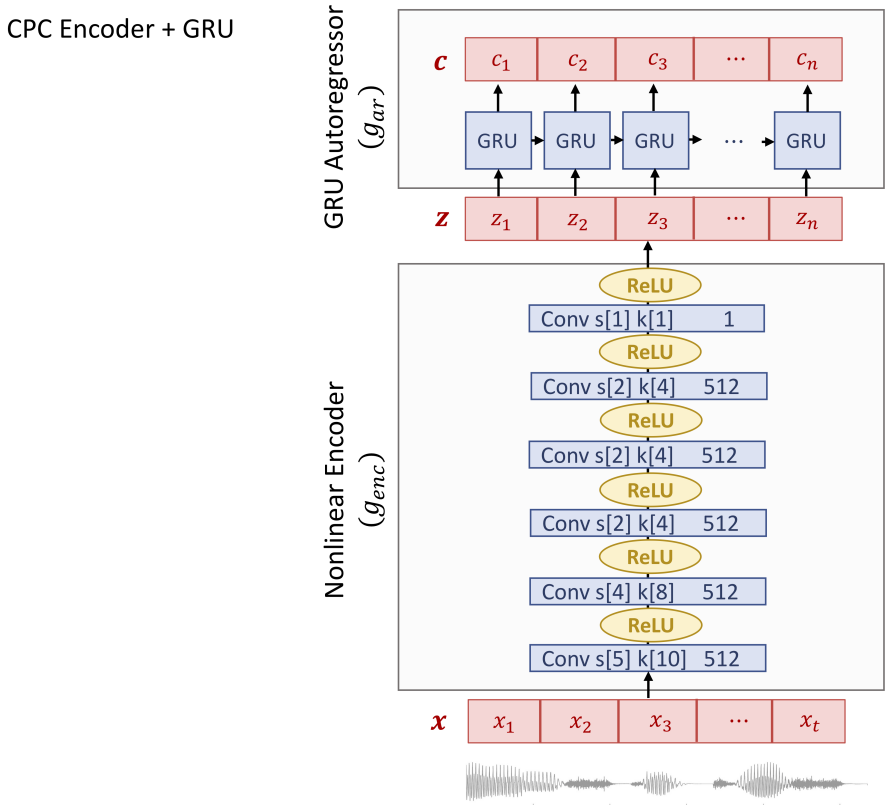

Figure 4: The convolutional encoder g_{enc} and GRU autoregressor g_{ar} architectures used to generate a latent representation z and context latent representation c for the $\text{CPC}_{\text{GRU}}+\text{WN}_{\text{dec}}$ model. g_{enc} consists of sequential 1-D convolutional layers with the listed stride, kernel sizes, sizes, and number of output channels. All activations were ReLU activations and no biases were used. The GRU layer has a 64-dimensional hidden state.

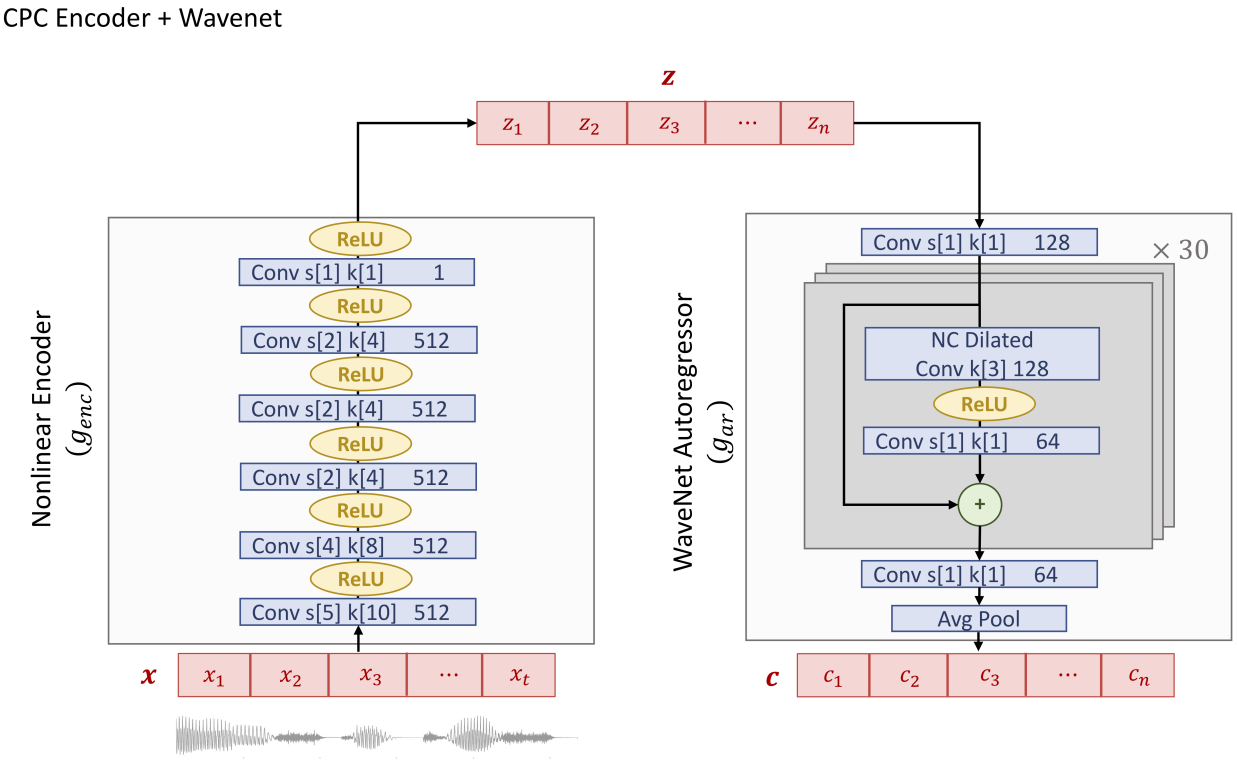

Figure 5: The convolutional encoder g_{enc} and WaveNet autoregressor g_{ar} architectures used to generate a latent representation z and context latent representation c for the $\text{CPC}_{\text{WN}}+\text{WN}_{\text{dec}}$ model. g_{enc} consists of sequential 1-D convolutional layers with the listed stride, kernel sizes, and number of output channels. All activations were ReLU activations and no biases were used. The WaveNet encoder consists of several dilated 1-D convolutions and traditional 1-D convolutions with skip connections. A final 1-D convolution and average pooling in the WaveNet encoder then produces our context latent sequence c_t.

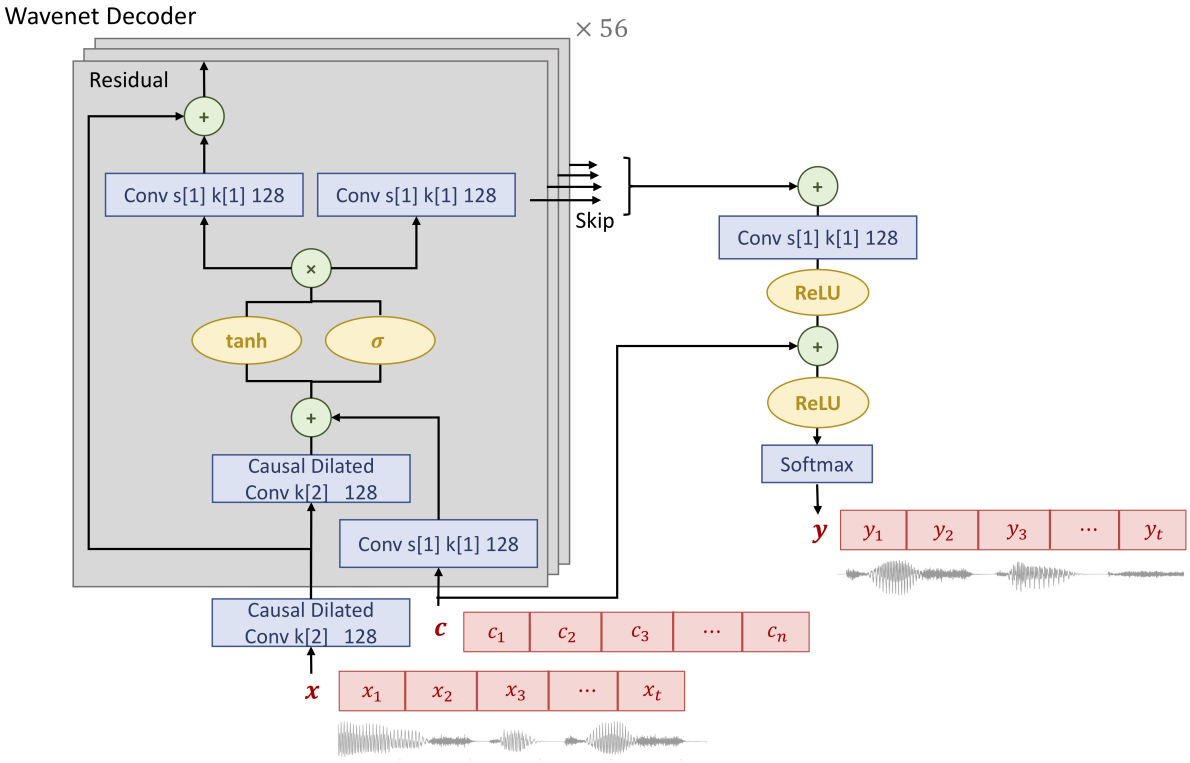

Figure 6: The WaveNet decoder created for each output domain. The architecture consists of traditional convolutions and dilated causal convolutions as described in the original WaveNet paper [7]. The corresponding context latent sequence c to a raw audio sequence x can be used to condition the decoder.

CDiscriminator

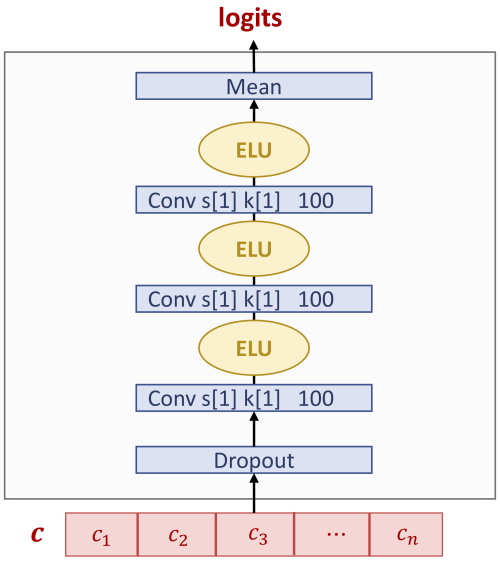

Figure 7: A domain confusion discriminator on the latent space discourages domain-specific embeddings. The architecture is exactly the same as the one used in UMT [2], which largely consists of sequential 1-D convolutional layers.