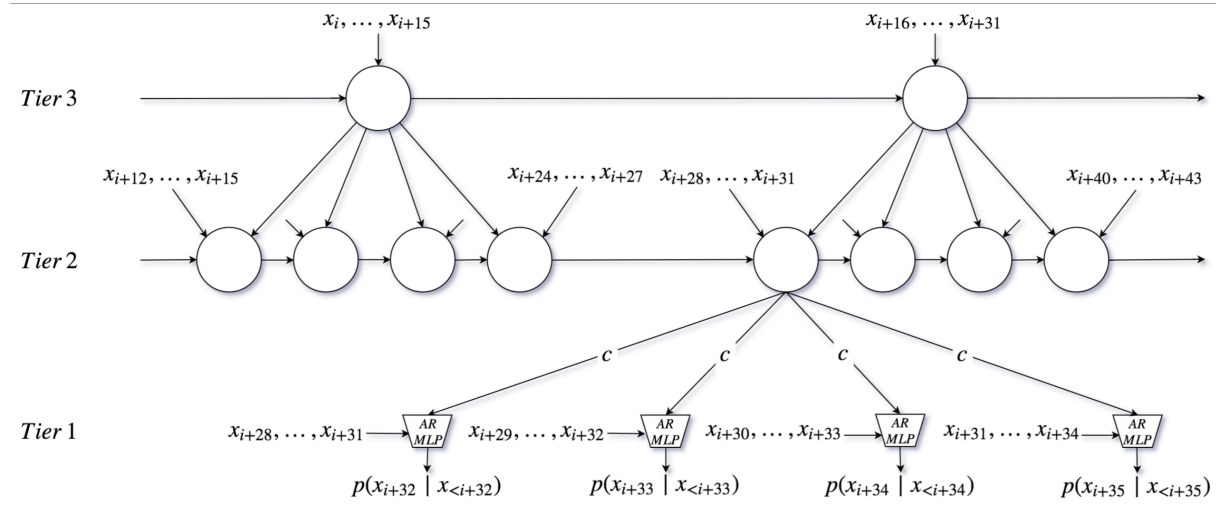


---

## SampleRNN

Code accompanying the paper [SampleRNN: An Unconditional End-to-End Neural Audio Generation Model](#) [here and here]. Samples are available [here](#).



## Dependencies

Extensively tested with: - cuDNN 5105 - Python 2.7.12 - Numpy 1.11.1 - Theano 0.8.2 (0.9 for WaveNet re-implementation) - Lasagne 0.2.dev1

## Datasets

Music dataset was created from all 32 Beethoven's piano sonatas available publicly on [archive.org](#). [datasets/music](#) contains scripts to preprocess and build this dataset. It is also available [here](#) for download. Extract the tar file and put all the numpy files in [datasets/music](#) directory.

## Training

To train a model on an existing dataset with accelerated GPU processing, you need to run following lines from the root of [sampleRNN\\_ICLR2017](#) folder which corresponds to the best found set of hyper-paramters.

Mission control center:

```
1 $ pwd
2 /u/mehris/sampleRNN_ICLR2017
```

---

## SampleRNN (2-tier)

```
1 $ python models/two_tier/two_tier.py -h
2 usage: two_tier.py [-h] [--exp EXP] --n_frames N_FRAMES --frame_size
3                     FRAME_SIZE --weight_norm WEIGHT_NORM --emb_size
4                     EMB_SIZE
5                     --skip_conn SKIP_CONN --dim DIM --n_rnn {1,2,3,4,5}
6                     --rnn_type {LSTM,GRU} --learn_h0 LEARN_H0 --q_levels
7                     Q_LEVELS --q_type {linear,a-law,mu-law} --which_set
8                     {ONOM,BLIZZ,MUSIC} --batch_size {64,128,256} [--
9                     debug]
10                    [--resume]
11
12 two_tier.py No default value! Indicate every argument.
13
14 optional arguments:
15   -h, --help            show this help message and exit
16   --exp EXP              Experiment name
17   --n_frames N_FRAMES   How many "frames" to include in each Truncated
18                         BPTT
19   --frame_size FRAME_SIZE
20                         pass
21                         How many samples per frame
22   --weight_norm WEIGHT_NORM
23                         Adding learnable weight normalization to all
24                         the
25                         linear layers (except for the embedding layer)
26   --emb_size EMB_SIZE   Size of embedding layer (0 to disable)
27   --skip_conn SKIP_CONN
28                         Add skip connections to RNN
29   --dim DIM              Dimension of RNN and MLPs
30   --n_rnn {1,2,3,4,5}   Number of layers in the stacked RNN
31   --rnn_type {LSTM,GRU}
32                         GRU or LSTM
33   --learn_h0 LEARN_H0   Whether to learn the initial state of RNN
34   --q_levels Q_LEVELS   Number of bins for quantization of audio
35                         samples.
36   --q_type {linear,a-law,mu-law}
37                         Should be 256 for mu-law.
38                         Quantization in linear-scale, a-law-companding,
39                         or mu-
40                         law compandig. With mu-/a-law quantization
41                         level shoud
42                         be set as 256
43   --which_set {ONOM,BLIZZ,MUSIC}
44                         ONOM, BLIZZ, or MUSIC
45   --batch_size {64,128,256}
46                         size of mini-batch
47   --debug                Debug mode
48   --resume                Resume the same model from the last checkpoint.
49   Order
```

To run:

```
1 $ THEANO_FLAGS=mode=FAST_RUN,device=gpu0,floatX=float32 python -u
  models/two_tier/two_tier.py --exp BEST_2TIER --n_frames 64 --
  frame_size 16 --emb_size 256 --skip_conn False --dim 1024 --n_rnn 3
  --rnn_type GRU --q_levels 256 --q_type linear --batch_size 128 --
  weight_norm True --learn_h0 True --which_set MUSIC
```

### SampleRNN (3-tier)

```
1 $ python models/three_tier/three_tier.py -h
2 usage: three_tier.py [-h] [--exp EXP] --seq_len SEQ_LEN --
  big_frame_size
3                        BIG_FRAME_SIZE --frame_size FRAME_SIZE --
  weight_norm
4                        WEIGHT_NORM --emb_size EMB_SIZE --skip_conn
  SKIP_CONN
5                        --dim DIM --n_rnn {1,2,3,4,5} --rnn_type {LSTM,GRU
  }
6                        --learn_h0 LEARN_H0 --q_levels Q_LEVELS --q_type
7                        {linear,a-law,mu-law} --which_set {ONOM,BLIZZ,
  MUSIC}
8                        --batch_size {64,128,256} [--debug] [--resume]
9
10 three_tier.py No default value! Indicate every argument.
11
12 optional arguments:
13   -h, --help            show this help message and exit
14   --exp EXP              Experiment name
15   --seq_len SEQ_LEN     How many samples to include in each Truncated
  BPTT
16                        pass
17   --big_frame_size BIG_FRAME_SIZE
18                        How many samples per big frame in tier 3
19   --frame_size FRAME_SIZE
20                        How many samples per frame in tier 2
21   --weight_norm WEIGHT_NORM
22                        Adding learnable weight normalization to all
  the
23                        linear layers (except for the embedding layer)
24   --emb_size EMB_SIZE   Size of embedding layer (> 0)
25   --skip_conn SKIP_CONN
26                        Add skip connections to RNN
27   --dim DIM              Dimension of RNN and MLPs
28   --n_rnn {1,2,3,4,5}   Number of layers in the stacked RNN
29   --rnn_type {LSTM,GRU}
30                        GRU or LSTM
```

---

```

31  --learn_h0 LEARN_H0    Whether to learn the initial state of RNN
32  --q_levels Q_LEVELS    Number of bins for quantization of audio
                           samples.
33                           Should be 256 for mu-law.
34  --q_type {linear,a-law,mu-law}
35                           Quantization in linear-scale, a-law-companding,
                           or mu-
36                           law compandig. With mu-/a-law quantization
                           level should
37                           be set as 256
38  --which_set {ONOM,BLIZZ,MUSIC}
39                           ONOM, BLIZZ, or MUSIC
40  --batch_size {64,128,256}
41                           size of mini-batch
42  --debug                Debug mode
43  --resume                Resume the same model from the last checkpoint.
                           Order
44                           of params are important. [for now]

```

To run:

```

1  $ THEANO_FLAGS=mode=FAST_RUN,device=gpu0,floatX=float32 python -u
   models/three_tier/three_tier.py --exp BEST_3TIER --seq_len 512 --
   big_frame_size 8 --frame_size 2 --emb_size 256 --skip_conn False --
   dim 1024 --n_rnn 1 --rnn_type GRU --q_levels 256 --q_type linear --
   batch_size 128 --weight_norm True --learn_h0 True --which_set MUSIC

```

## Reference

If you are using this code, please cite the paper.

```

@article{mehri2016samplernn, Author = {Soroush Mehri and Kundan Kumar
and Ishaan Gulrajani and Rithesh Kumar and Shubham Jain and Jose
Sotelo and Aaron Courville and Yoshua Bengio}, Title = {SampleRNN:
An Unconditional End-to-End Neural Audio Generation Model}, Year =
{2016}, Journal = {arXiv preprint arXiv:1612.07837}, }

```

## Torch implementation

Thanks to Richard Assar, now we have a Torch implementation available:

[https://github.com/richardassar/SampleRNN\\_torch](https://github.com/richardassar/SampleRNN_torch)

---

## **Miscellaneous**

- Talk by Yoshua Bengio at CBMM, MIT: Deep Generative Models for Speech and Images
- Follow-up project: Char2Wav: End-To-End Speech Synthesis

If needed or have interesting related project/results, please don't hesitate to contact us.