The NSynth Dataset

Apr 5, 2017 • NSynth

A large-scale and high-quality dataset of annotated musical notes.

Download

Contents

- How to Cite
- Updates

Motivation

Recent breakthroughs in generative modeling of images have been predicated on the availability of high-quality and large-scale datasebts such as MNIST, CIFAR and ImageNet. We recognized the need for an audio dataset that was as approachable as those in the image domain.

Audio signals found in the wild contain multi-scale dependencies that prove particularly difficult to model, leading many previous efforts at data-driven audio synthesis to focus on more constrained domains such as texture synthesis or training small parametric models.

We encourage the broader community to use NSynth as a benchmark and entry point into audio machine learning. We also view NSynth as a building block for future datasets and envision a high-quality multi-note dataset for tasks like generation and transcription that involve learning complex language-like dependencies.

Description

NSynth is an audio dataset containing 305,979 musical notes, each with a unique pitch, timbre, and envelope. For 1,006 instruments from commercial sample libraries, we generated four second, monophonic 16kHz audio snippets, referred to as notes, by ranging over every pitch of a standard MIDI pian o (21-108) as well as five different velocities (25, 50, 75, 100, 127). The note was held for the first three seconds and allowed to decay for the final second.

Some instruments are not capable of producing all 88 pitches in this range, resulting in an average of 65.4 pitches per instrument. Furthermore, the commercial sample packs occasionally contain duplicate sounds across multiple velocities, leaving an average of 4.75 unique velocities per pitch.

We also annotated each of the notes with three additional pieces of information based on a combination of human evaluation and heuristic algorithms:

- *Source*: The method of sound production for the note's instrument. This can be one of `acoustic` or `electronic` for instruments that were recorded from acoustic or electronic instruments, respectively, or `synthetic` for synthesized instruments. See their frequencies below.

- *Family*: The high-level family of which the note's instrument is a member. Each instrument is a member of exactly one family. See the complete list and their frequencies below.

- *Qualities*: Sonic qualities of the note. See the quality descriptions and their co-occurrences below. Each note is annotated with zero or more qualities.

Format

Files

The NSynth dataset can be download in two formats:

- TFRecord files of serialized TensorFlow Example protocol buffers with one Example proto per note.
- JSON files containing non-audio features alongside 16-bit PCM WAV audio files.

The full dataset is split into three sets:

- **Train** [tfrecord | json/wav]: A training set with 289,205 examples. Instruments do not overlap with valid or test.
- **Valid** [tfrecord | json/wav]: A validation set with 12,678 examples. Instruments do not overlap with train.
- **Test** [tfrecord | json/wav]: A test set with 4,096 examples. Instruments do not overlap with train.

Below we detail how the note features are encoded in the Example protocol buffers and JSON files.

Example Features

Each Example contains the following features.

Feature	Type	Description
note	int64	A unique integer identifier for the note.
note_str	bytes	A unique string identifier for the note in the format <instrument_str>-<pitch>-<velocity> .
instrument	int64	A unique, sequential identifier for the instrument the note was synthesized from.
instrument_str	bytes	A unique string identifier for the instrument this note was synthesized from in the format <instrument_family_str>-<instrument_production_str>-<instrument_name> .
pitch	int64	The 0-based MIDI pitch in the range [0, 127].
velocity	int64	The 0-based MIDI velocity in the range [0, 127].
sample_rate	int64	The samples per second for the audio feature.
audio*	[float]	A list of audio samples represented as floating point values in the range [-1,1].
qualities	[int64]	A binary vector representing which sonic qualities are present in this note.

Feature	Type	Description
qualities_str	[bytes]	A list IDs of which qualities are present in this note selected from the sonic qualities list.
instrument_family	int64	The index of the instrument family this instrument is a member of.
instrument_family_str	bytes	The ID of the instrument family this instrument is a member of.
instrument_source	int64	The index of the sonic source for this instrument.
instrument_source_str	bytes	The ID of the sonic source for this instrument.

* **Note**: the "audio" feature is ommited from the JSON-encoded examples since the audio data is stored separately in WAV files keyed by the "note_str".

Feature Encodings

This section includes tables specifying the feature names and indicies used in the Example protos.

Instrument Sources

The method of sound production for the note's instrument. Each instrument (and all of its notes) are labeled with exactly one.

Index	ID
0	acoustic
1	electronic

Index	ID
2	synthetic

Instrument Families

The high-level family of which the note’s instrument is a member. Each instrument (and all of its notes) are labeled with exactly one.

Index	ID
0	bass
1	brass
2	flute
3	guitar
4	keyboard
5	mallet
6	organ
7	reed
8	string
9	synth_lead

Index	ID
10	vocal

Note Qualities

We provide quality annotations for the 10 different note qualities described below. None of the tags are mutually exclusive by definition except for “bright” and “dark”. However, it is possible for a note to be neither “bright” nor “dark”.

Index	ID	Description
0	bright	A large amount of high frequency content and strong upper harmonics.
1	dark	A distinct lack of high frequency content, giving a muted and bassy sound. Also sometimes described as ‘Warm’.
2	distortion	Waveshaping that produces a distinctive crunchy sound and presence of many harmonics. Sometimes paired with non-harmonic noise.
3	fast_decay	Amplitude envelope of all harmonics decays substantially before the ‘note-off’ point at 3 seconds.
4	long_release	Amplitude envelope decays slowly after the ‘note-off’ point, sometimes still present at the end of the sample 4 seconds.
5	multiphonic	Presence of overtone frequencies related to more than one fundamental frequency.

Index	ID	Description
6	nonlinear_env	Modulation of the sound with a distinct envelope behavior different than the monotonic decrease of the note. Can also include filter envelopes as well as dynamic envelopes.
7	percussive	A loud non-harmonic sound at note onset.
8	reverb	Room acoustics that were not able to be removed from the original sample.
9	tempo-synced	Rhythmic modulation of the sound to a fixed tempo.

Example

Below is a string view of the Example protocol buffer for an individual note in the dataset, with the audio portion suppressed:

```
{    # (tensorflow.Example) size=250.4K
  features: {   # (tensorflow.Features) size=250.4K
    feature: {  # (tensorflow.Features.FeatureEntry) size=21B
      key  : "sample_rate"      # size=11
      value: {  # (tensorflow.Feature) size=6B
        int64_list: {    # (tensorflow.Int64List) size=4B
          value: [ 16000 ]
        }       # features.feature[0].value.int64_list
      } # features.feature[0].value
    }  # features.feature[0]
    feature: {  # (tensorflow.Features.FeatureEntry) size=25B
      key  : "qualities_str"    # size=13
      value: {  # (tensorflow.Feature) size=8B
        bytes_list: {   # (tensorflow.BytesList) size=6B
          value: [ "dark" ]
        }       # features.feature[1].value.bytes_list
      } # features.feature[1].value
    }   # features.feature[1]
    feature: {  # (tensorflow.Features.FeatureEntry) size=42B
      key  : "note_str" # size=8
      value: {  # (tensorflow.Feature) size=30B
        bytes_list: {   # (tensorflow.BytesList) size=28B
          value: [ "bass_synthetic_033-022-050" ]      # size=26
        }       # features.feature[2].value.bytes_list
      } # features.feature[2].value
    }   # features.feature[2]
    feature: {  # (tensorflow.Features.FeatureEntry) size=27B
      key  : "qualities"        # size=9
      value: {  # (tensorflow.Feature) size=14B
        int64_list: {   # (tensorflow.Int64List) size=12B
          value: [ 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 ]
        }       # features.feature[3].value.int64_list
      } # features.feature[3].value
    }   # features.feature[3]
    feature: {  # (tensorflow.Features.FeatureEntry) size=250.0K
      key  : "audio"    # size=5
      value: {  # (tensorflow.Feature) size=250.0K
        float_list: {   # (tensorflow.FloatList) size=250.0K
          value: [ -1.3311218e-07, ...,  1.3244664e-07 ]
        }       # features.feature[4].value.float_list
      } # features.feature[4].value
    }   # features.feature[4]
    feature: {  # (tensorflow.Features.FeatureEntry) size=26B
```

```
 key  : "instrument_family"      # size=17
 value: {  # (tensorflow.Feature) size=5B
  int64_list: {   # (tensorflow.Int64List) size=3B
   value: [ 0 ]
  }       # features.feature[5].value.int64_list
 } # features.feature[5].value
}  # features.feature[5]
feature: {  # (tensorflow.Features.FeatureEntry) size=14B
 key  : "pitch"     # size=5
 value: {  # (tensorflow.Feature) size=5B
  int64_list: {   # (tensorflow.Int64List) size=3B
   value: [ 22 ]
  }       # features.feature[6].value.int64_list
 } # features.feature[6].value
}  # features.feature[6]
feature: {  # (tensorflow.Features.FeatureEntry) size=26B
 key  : "instrument_source"      # size=17
 value: {  # (tensorflow.Feature) size=5B
  int64_list: {   # (tensorflow.Int64List) size=3B
   value: [ 2 ]
  }       # features.feature[7].value.int64_list
 } # features.feature[7].value
}  # features.feature[7]
feature: {  # (tensorflow.Features.FeatureEntry) size=40B
 key  : "instrument_str"    # size=14
 value: {  # (tensorflow.Feature) size=22B
  bytes_list: {   # (tensorflow.BytesList) size=20B
   value: [ "bass_synthetic_033" ]       # size=18
  }       # features.feature[8].value.bytes_list
 } # features.feature[8].value
}  # features.feature[8]
feature: {  # (tensorflow.Features.FeatureEntry) size=38B
```

```
 key  : "instrument_source_str"    # size=21
 value: {  # (tensorflow.Feature) size=13B
  bytes_list: {   # (tensorflow.BytesList) size=11B
   value: [ "synthetic" ]       # size=9
  }       # features.feature[9].value.bytes_list
 } # features.feature[9].value
}  # features.feature[9]
feature: {  # (tensorflow.Features.FeatureEntry) size=15B
 key  : "note"
 value: {  # (tensorflow.Feature) size=7B
  int64_list: {   # (tensorflow.Int64List) size=5B
   value: [ 201034 ]      # 196.32Ki; [if seconds]: 2 days 7 hours
  }       # features.feature[10].value.int64_list
 } # features.feature[10].value
}  # features.feature[10]
feature: {  # (tensorflow.Features.FeatureEntry) size=20B
 key  : "instrument"       # size=10
 value: {  # (tensorflow.Feature) size=6B
  int64_list: {   # (tensorflow.Int64List) size=4B
   value: [ 417 ]
  }       # features.feature[11].value.int64_list
 } # features.feature[11].value
}  # features.feature[11]
feature: {  # (tensorflow.Features.FeatureEntry) size=33B
 key  : "instrument_family_str"    # size=21
 value: {  # (tensorflow.Feature) size=8B
  bytes_list: {   # (tensorflow.BytesList) size=6B
   value: [ "bass" ]
  }       # features.feature[12].value.bytes_list
 } # features.feature[12].value
}  # features.feature[12]
feature: {  # (tensorflow.Features.FeatureEntry) size=17B
```

```
   key  : "velocity" # size=8
   value: {  # (tensorflow.Feature) size=5B
    int64_list: {   # (tensorflow.Int64List) size=3B
     value: [ 50 ]
    }      # features.feature[13].value.int64_list
   } # features.feature[13].value
  }  # features.feature[13]
 }    # features
}
```

This is a view of the same Example in JSON format:

```
"bass_synthetic_033-022-050": {
  "note": 201034,
  "sample_rate": 16000,
  "instrument_family": 0,
  "qualities": [
    0,
    1,
    0,
    0,
    0,
    0,
    0,
    0,
    0,
    0
  ],
  "instrument_source_str": "synthetic",
  "note_str": "bass_synthetic_033-022-050",
  "instrument_family_str": "bass",
    "instrument_str": "bass_synthetic_033",
    "pitch": 22,
    "instrument": 417,
    "velocity": 50,
    "instrument_source": 2,
    "qualities_str": [
        "dark"
    ]
}
```

Statistics

Instrument Classes

Frequency counts for instrument classes with Sources as columns an Families as rows.

Family	Acoustic	Electronic	Synthetic	Total
Bass	200	8,387	60,368	68,955
Brass	13,760	70	0	13,830
Flute	6,572	35	2,816	9,423
Guitar	13,343	16,805	5,275	35,423
Keyboard	8,508	42,645	3,838	54,991
Mallet	27,722	5,581	1,763	35,066
Organ	176	36,401	0	36,577

Family	**Acoustic**	**Electronic**	**Synthetic**	**Total**
Reed	14,262	76	528	14,866
String	20,510	84	0	20,594
Synth Lead	0	0	5,501	5,501
Vocal	3,925	140	6,688	10,753
Total	108,978	110,224	86,777	305,979

Qualitiy Co-occurrences

Co-occurrence probabilities and marginal frequencies of quality annotations. Both are presented as percentages.

Quality	**Bright**	**Dark**	**Distortion**	**Fast Decay**	**Long Release**	**Multiphonic**	**Nonlinear Envelope**	**Percussive**	**Reverb**	**Tempo-Synced**
Dark	0.0									
Distortion	25.9	2.5								
Fast Decay	10.0	7.5	8.1							
Long Release	9.0	5.2	9.8	0.0						
Multiphonic	6.0	1.5	5.4	2.8	6.9					

Quality	**Bright**	**Dark**	**Distortion**	**Fast Decay**	**Long Release**	**Multiphonic**	**Nonlinear Envelope**	**Percussive**	**Reverb**	**Tempo-Synced**
Nonlinear Envelope	8.5	1.4	6.6	2.1	6.7	8.6				
Percussive	6.2	5.1	3.0	52.0	0.8	2.4	0.9			
Reverb	6.6	8.9	0.3	13.0	13.7	0.7	3.5	12.4		
Tempo-Synced	2.4	1.8	5.2	0.4	6.4	9.3	2.3	1.5	0.0	
Frequency	13.5	11.0	17.0	14.7	8.5	3.4	3.2	10.2	16.8	1.8

License

The dataset is made available by Google Inc. under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.

How to Cite

If you use the NSynth dataset in your work, please cite the paper where it was introduced:

```
Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck,
  Karen Simonyan, and Mohammad Norouzi. "Neural Audio Synthesis of Musical Notes
  with WaveNet Autoencoders." 2017.
```

You can also use the following BibTeX entry:

```
@misc{nsynth2017,
    Author = {Jesse Engel and Cinjon Resnick and Adam Roberts and
              Sander Dieleman and Douglas Eck and Karen Simonyan and
              Mohammad Norouzi},
    Title = {Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders},
    Year = {2017},
    Eprint = {arXiv:1704.01279},
}
```

Updates

- 04-10-2017: Removed 64 duplicate notes from train set.