https://bit.ly/flucoma-upenn

FluCoMa

Ted Moore

tedmooremusic.com

ted@tedmooremusic.com

FluCoMa: Fluid Corpus Manipulation

- “*enable techno-fluent musicians to use machine listening and machine learning in their creative practices*”
- Integrating Machine Listening and Machine Learning in…
- Max, SuperCollider & Pure Data
- Learning Resources (learn.flucoma.org)
- Discourse Community (discourse.flucoma.org)

Ted Moore, (James Bradbury, Owen Green, Jacob Hart, Gerard Roma, & Pierre Alexandre Tremblay)
University of Huddersfield | UK

FluCoMa: Fluid Corpus Manipulation

Slice Audio
- onset slice
- transient slice
- novelty slice
- amplitude slice
- amplitude gate

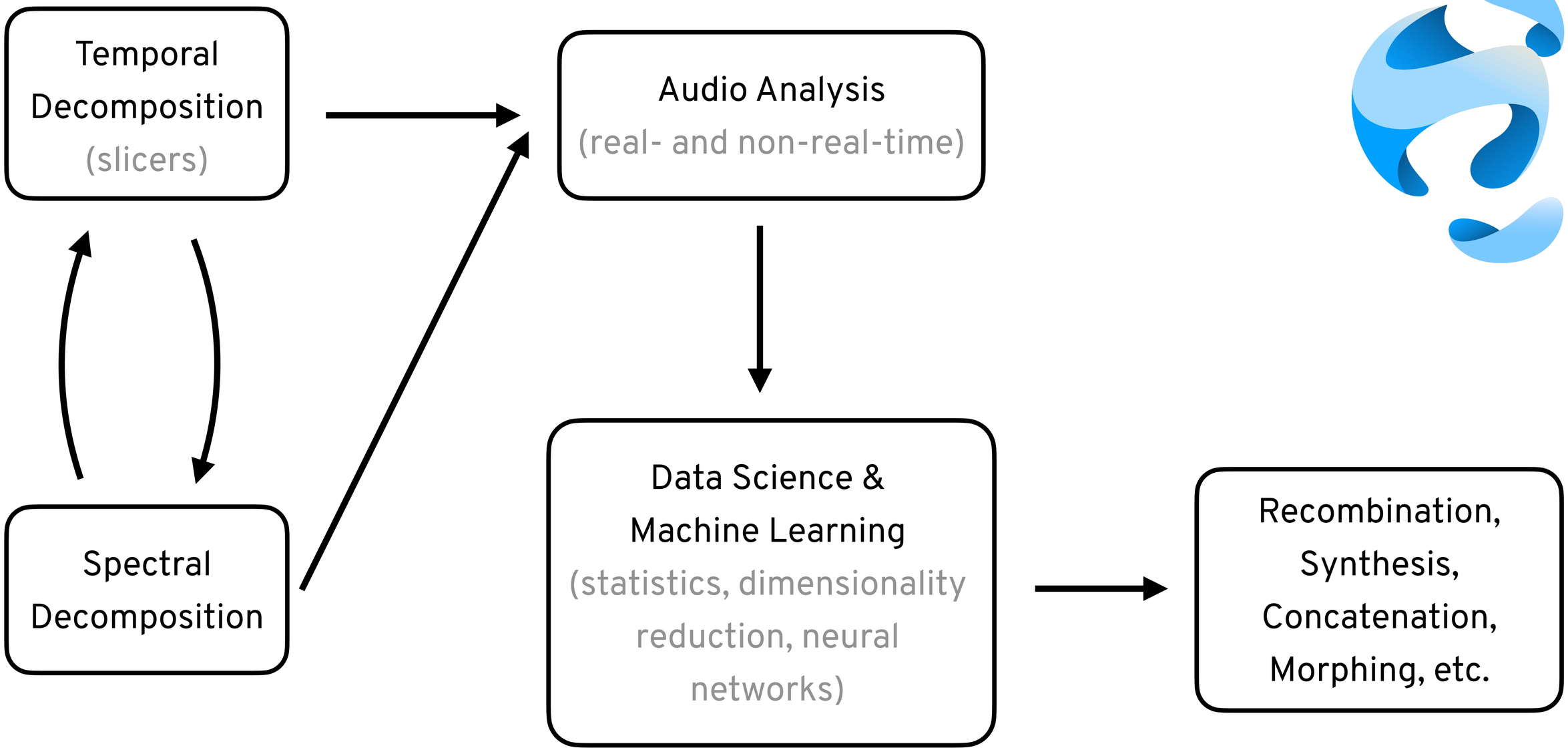
Decompose Audio
- extract transients
- harmonic/percussive separation
- model as sine waves
- non-negative matrix factorisation

Analyse Audio
- pitch
- loudness
- mel-bands
- mel-frequency cepstral coefficients
- spectral centroid
- spectral flatness
- chromagram

Transform Audio
- audio transport
- non-negative matrix factorisation filters & morphing

Analyse Data
- datasets
- labelsets
- statistical analyses
- normalization
- standardization
- robust scaler
- principal component analysis
- MDS
- KDTree
- K Nearest Neighbours
- neural networks
- SQL-type query
- KMeans
- UMAP
- grid

… and more


Temporal Decomposition (slicers)
Audio Analysis (real- and non-real-time)
Spectral Decomposition
Data Science & Machine Learning (statistics, dimensionality reduction, neural networks)
Recombination, Synthesis, Concatenation, Morphing, etc.

Plan for the Week...

FluidMLPClassifier

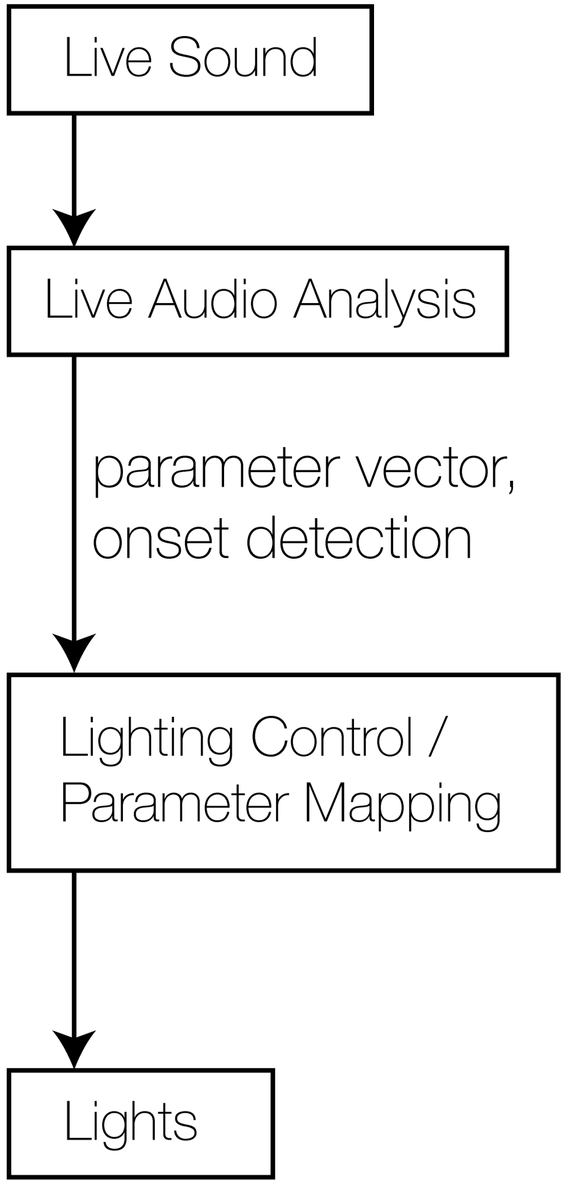
classify sounds by
timbre in real-time

feed

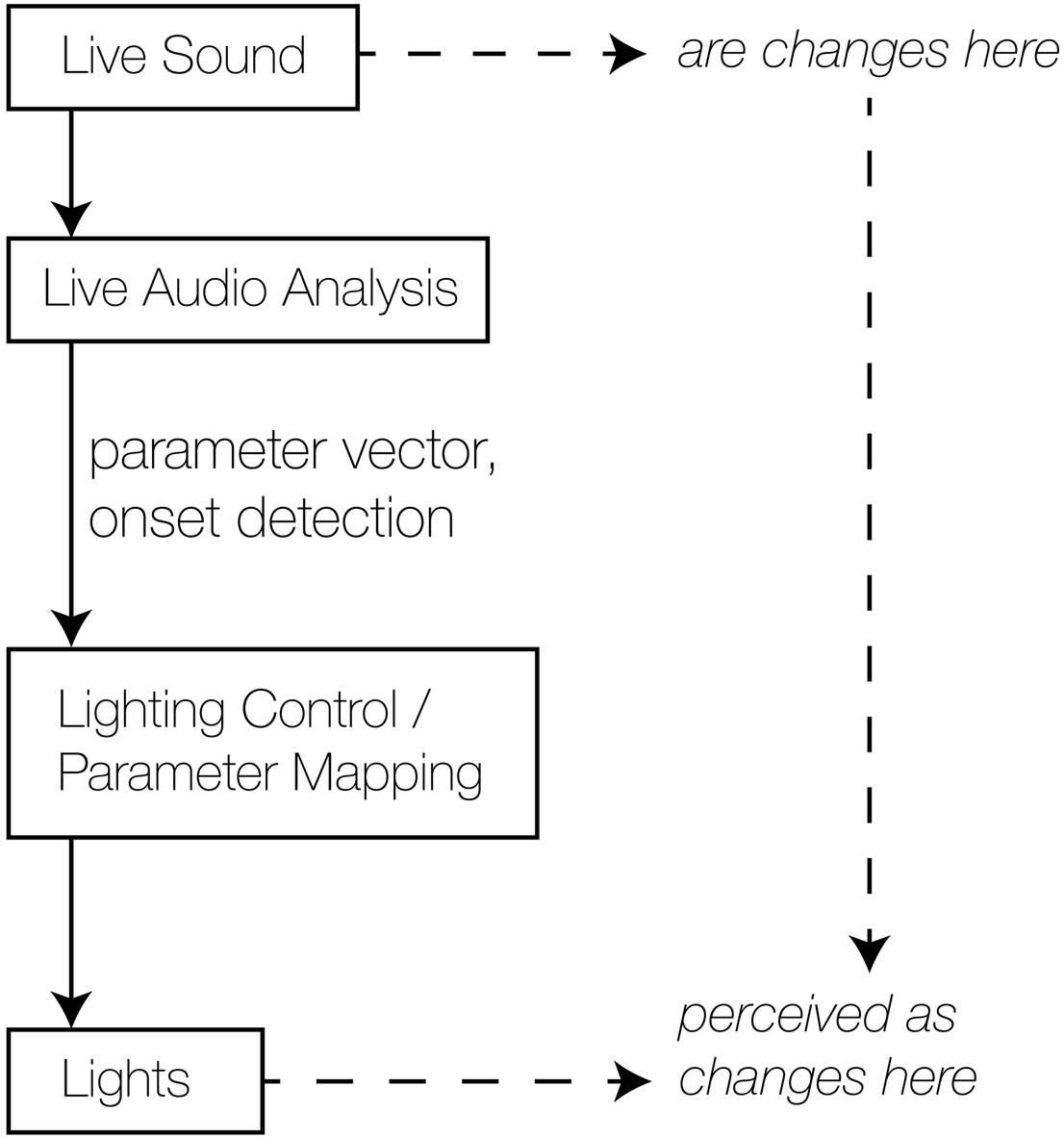
using a neural network for real-time audio classification

training a machine to hear the way I hear

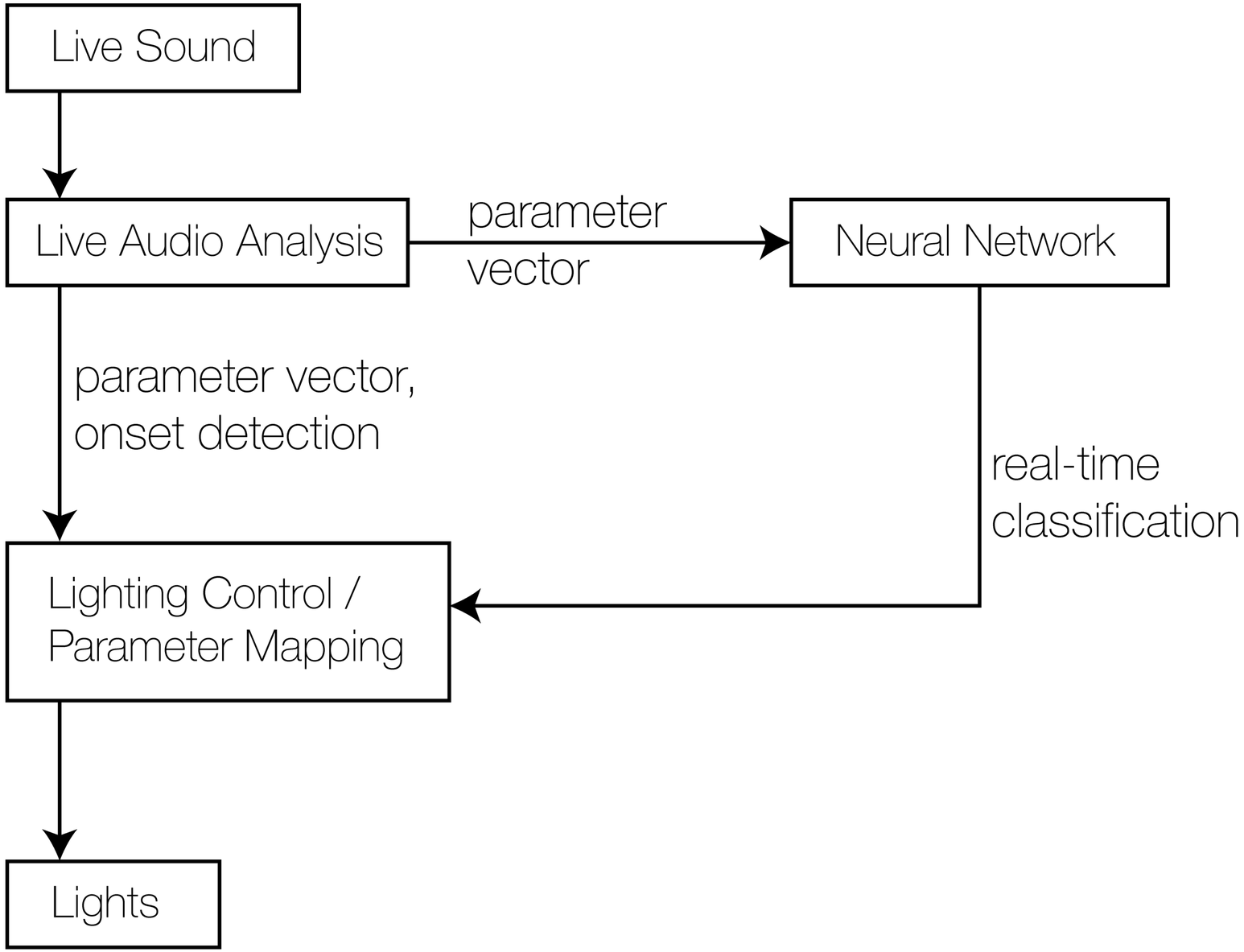
Machine Listening System

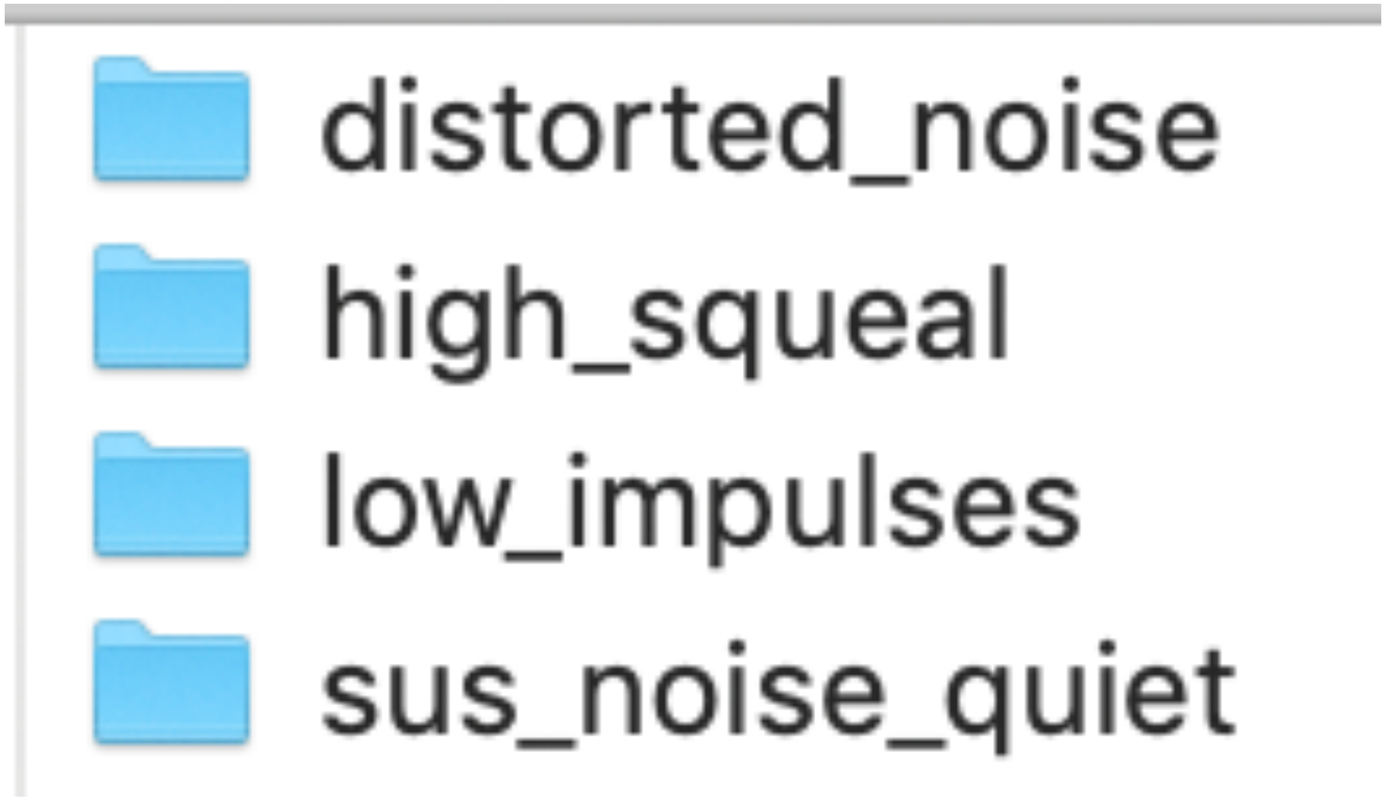


Machine Listening System



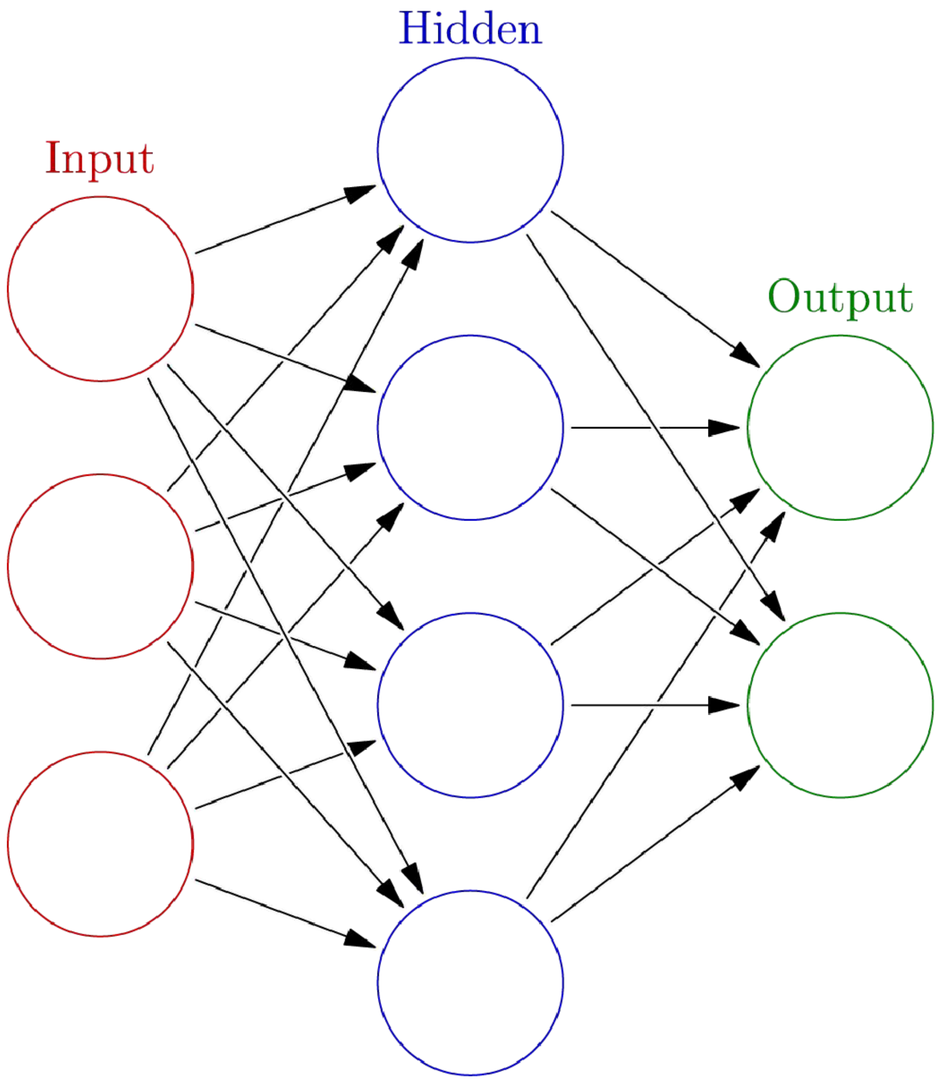
Machine Learning System




distorted_noise
high_squeal
low_impulses
sus_noise_quiet

flashing lights warning

Neural Network (Multi-Layer Perceptron)



<u>Classification:</u>

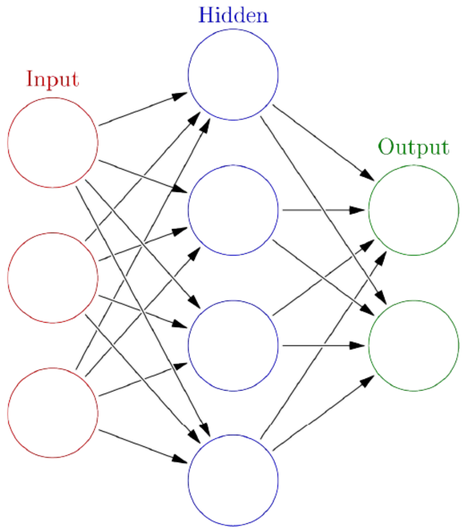
a neural network predicts

which category (or “class”)

an input belongs to

Neural Network *Training a Classifier*

trombone examples	oboe examples



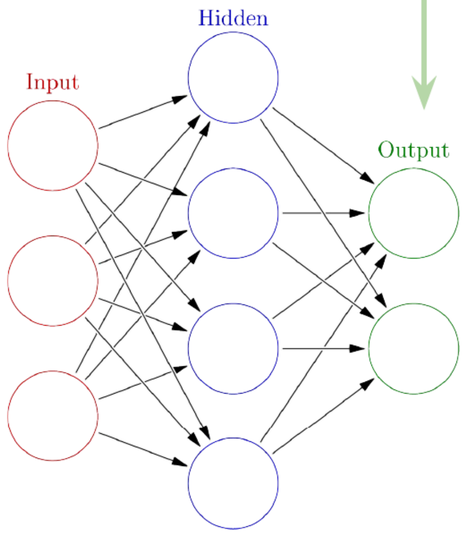
Neural Network *Training a Classifier*

trombone examples

oboe examples

"this example is a trombone"

audio analysis of example (e.g., pitch, brightness, noisiness)



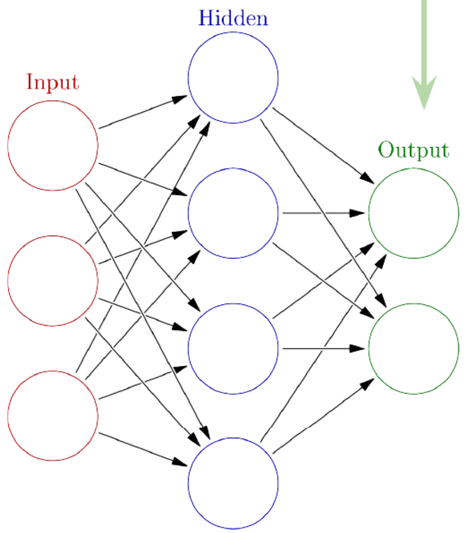
Neural Network *Training a Classifier*

trombone examples

oboe examples

"this example is an oboe"

audio analysis of example (e.g., pitch, brightness, noisiness)



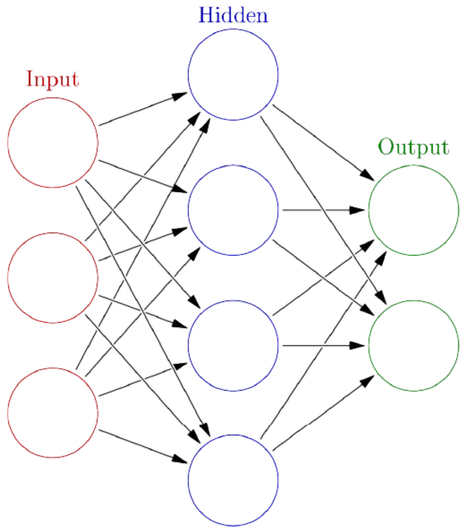
Neural Network *Predicting a Classification*

trombone examples

oboe examples

new example it has never seen before

audio analysis of example (e.g., pitch, brightness, noisiness)



“this new example is most like the oboe examples you showed me before”

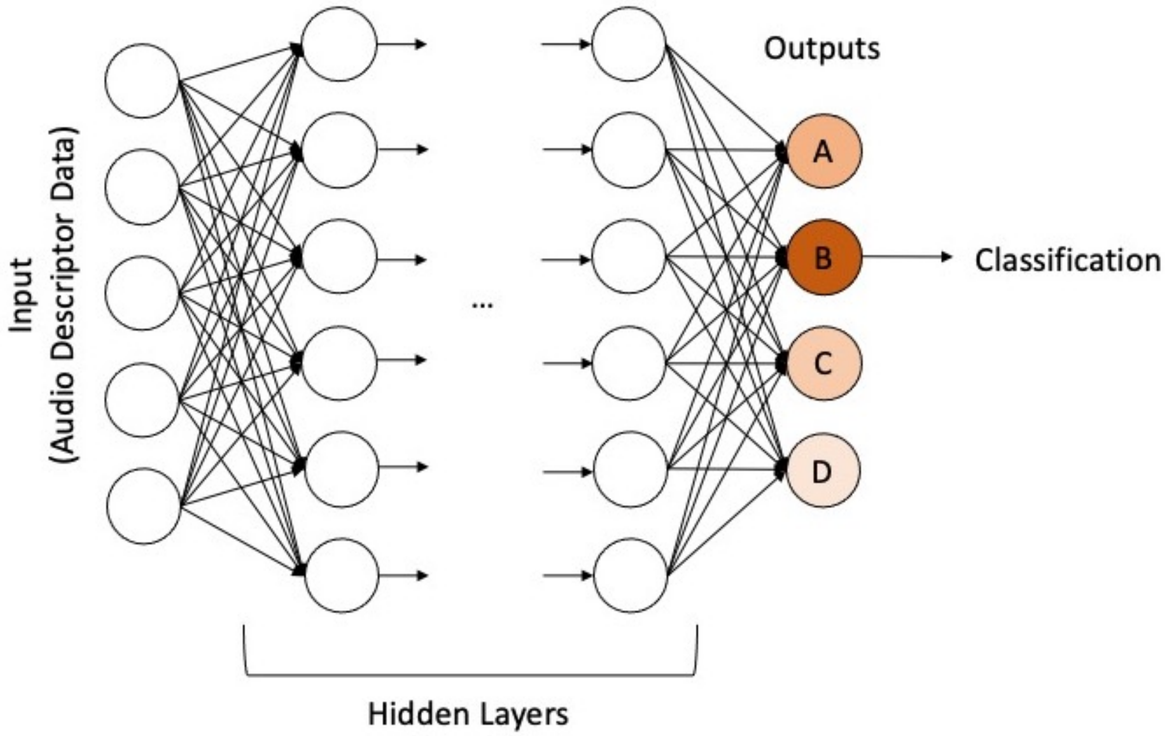
(or trombone…)

fluid.mlpclassifier~

fluid.mlpclassifier~

validation

- **overfitting**
  - excellent on training data, poor on anything else
- **underfitting**
  - poor on everything…keep training!



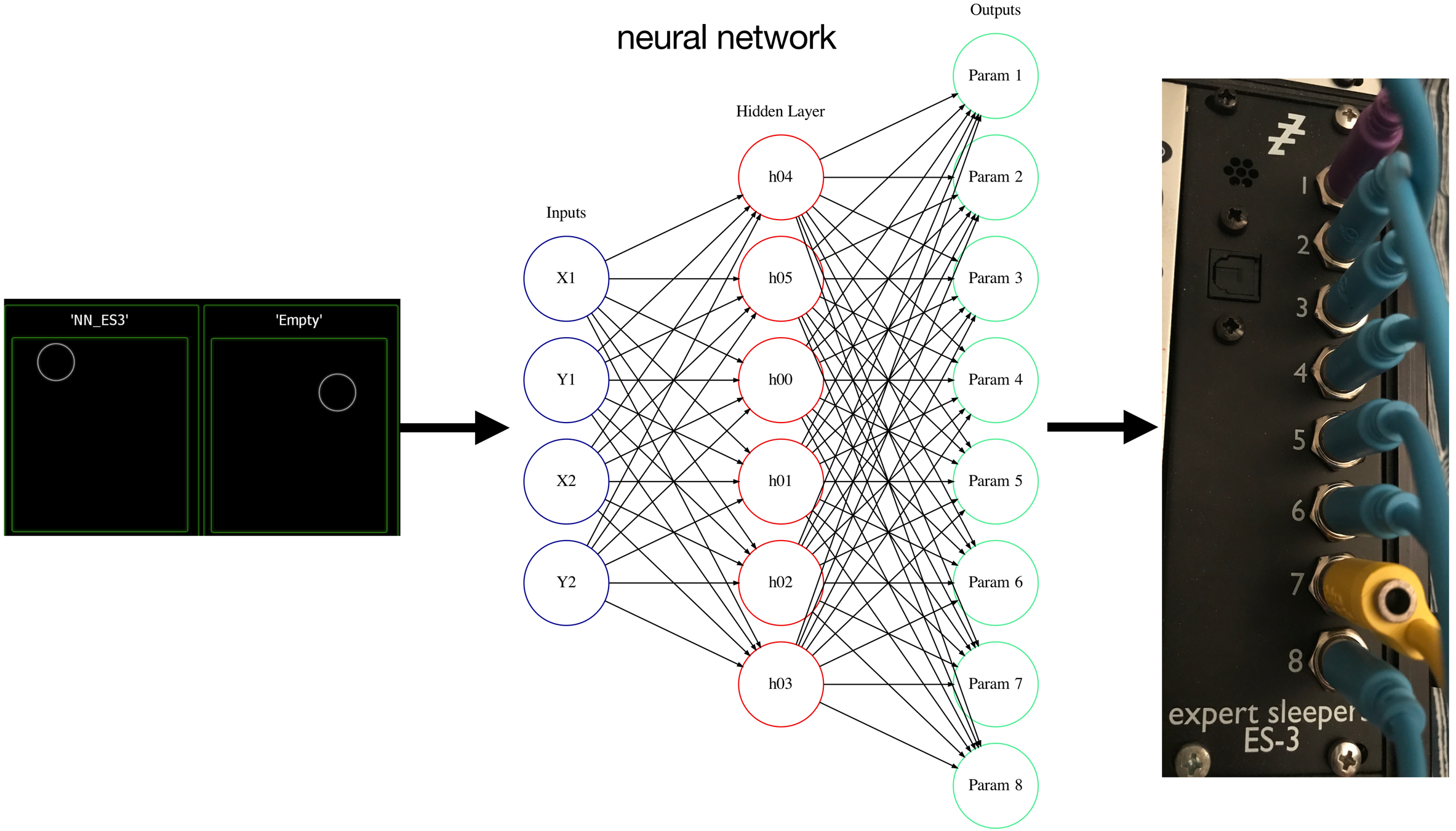
Drift Shadow
by Alex Harker

learn.flucoma.org/explore/harker

MLPRegressor

control *many* synthesizer parameters from a smaller control space

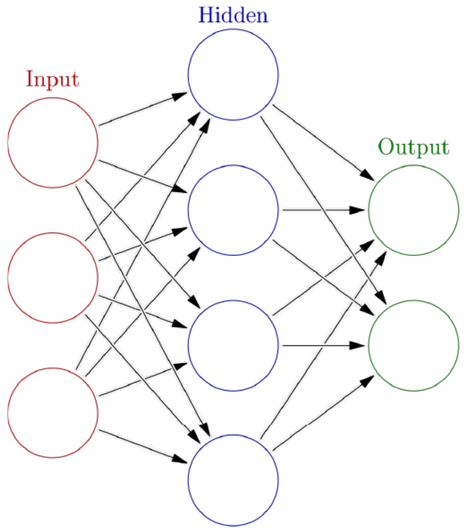

KORG
nanoKONTROL2
TRACK
MARKER
CYCLE
SET
S
M
R
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8
Keith McMillen
QuNeo
0
1
2
3
SPEARPlayer
PShiftDel
Random Delay
SpecSmear
PitchShift
Benjolin
AmpMod


neural network
'NN_ES3'
'Empty'
Inputs
X1
Y1
X2
Y2
Hidden Layer
h04
h05
h00
h01
h02
h03
Outputs
Param 1
Param 2
Param 3
Param 4
Param 5
Param 6
Param 7
Param 8
1
2
3
4
5
6
7
8
expert sleepers
ES-3


Slingerland

Neural Network *Training a Regressor*

identifier	input examples		output examples
“a”:	[0.1, 0.5, 1]	→	[0.5, 0.7]
“b”:	[0, 1, 1]	→	[0.1, 0.1]
“c”:	[1, 0, 1]	→	[0.5, 0.6]
“d”:	[0.2, 0, 1]	→	[0.3, 0.9]
“e”:	[0, 0, 0.5]	→	[0.4, 0]
“f”:	[0.8, 0.2, 1]	→	[0.2, 0.7]
“g”:	[0.9, 0.3, 0]	→	[0.5, 0.6]
“h”:	[0.4, 1, 0.7]	→	[0.6, 0.1]
“i”:	[0.2, 0.3, 1]	→	[0.5, 0.3]
“j”:	[0.1, 1, 0.1]	→	[0.5, 0.7]
“k”:	[0, 1, 1]	→	[0.1, 0.8]
“l”:	[0, 1, 1]	→	[0.5, 0.1]
“m”:	[0, 0, 1]	→	[0.4, 0.2]
“n”:	[0, 0, 1]	→	[0.1, 0.8]



Neural Network *Training a Regressor*

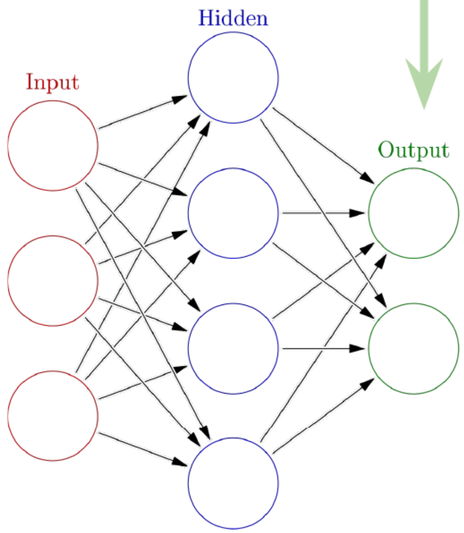
identifier

input examples

output examples

"please return these output values"

"when I input these values…"



Makes a guess on every example and determines how it was wrong. Improves a bit every time.

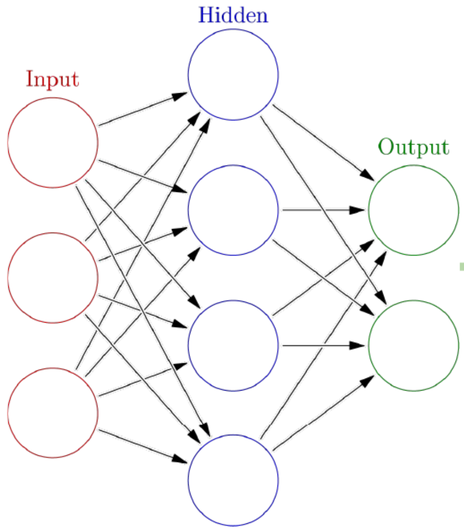
Neural Network *Predicting with Regression*

input examples

output examples

“here are some values you’ve maybe never seen before”

[0.6, 0.2, 0.4]



“here’s an output that would ‘fit’ in your training data”

[0.9, 0.8]

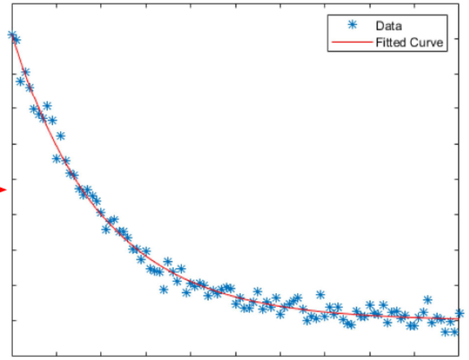
Neural Network *Predicting with Regression*

input examples

output examples

"here are some values you've maybe never seen before"

[0.6, 0.2, 0.4]



"here's an output that would 'fit' in your training data"

[0.9, 0.8]

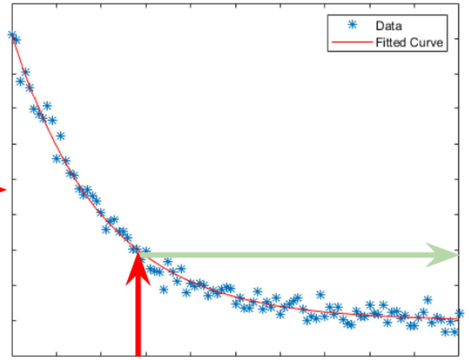
Neural Network *Predicting with Regression*

input examples

output examples

"here are some values you've maybe never seen before"

[0.6, 0.2, 0.4]

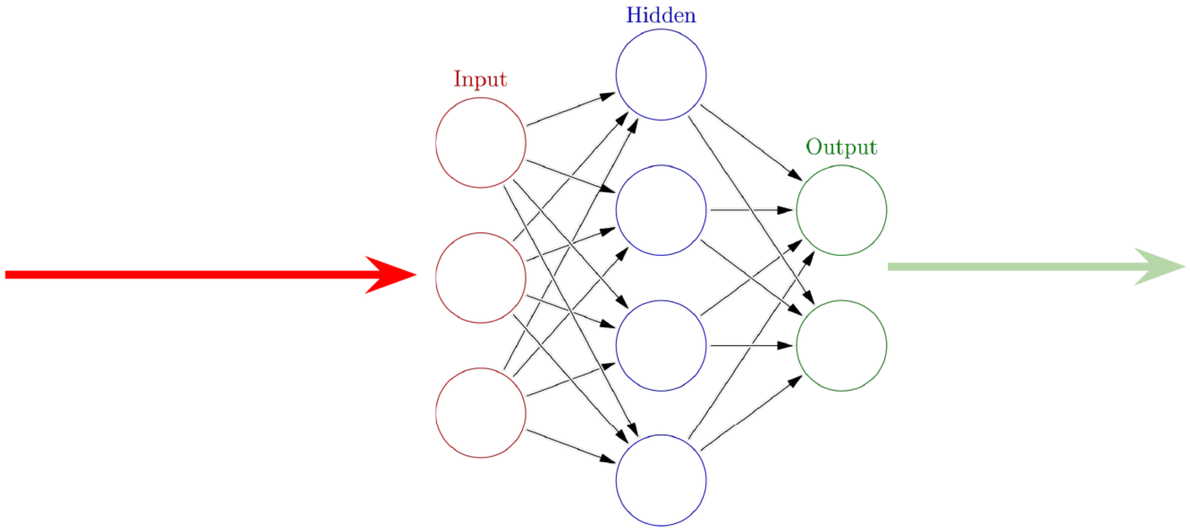


"here's an output that would 'fit' in your training data"

[0.9, 0.8]

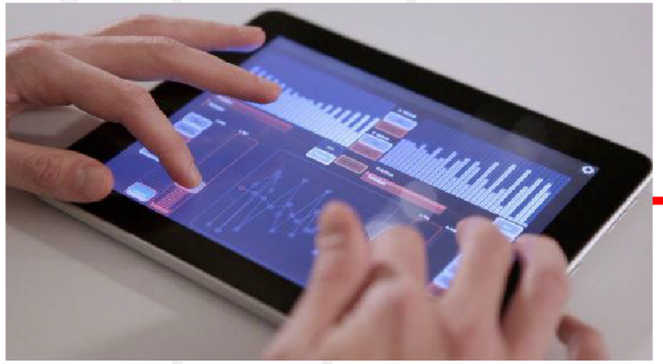
Neural Network *Predicting with Regression*

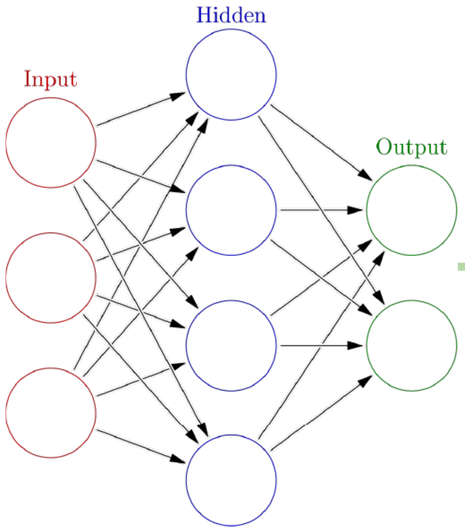
"a": [0.1, 0.5, 1]
"b": [0, 1, 1]
"c": [1, 0, 1]
"d": [0.2, 0, 1]
"e": [0, 0, 0.5]
"f": [0.8, 0.2, 1]
"g": [0.9, 0.3, 0]
"h": [0.4, 1, 0.7]
"i": [0.2, 0.3, 1]
"j": [0.1, 1, 0.1]
"k": [0, 1, 1]
"l": [0, 1, 1]
"m": [0, 0, 1]
"n": [0.4, 1, 0.7]
"o": [0.2, 0.3, 1]
"p": [0.1, 1, 0.1]
"q": [0, 1, 1]
"r": [0, 1, 1]

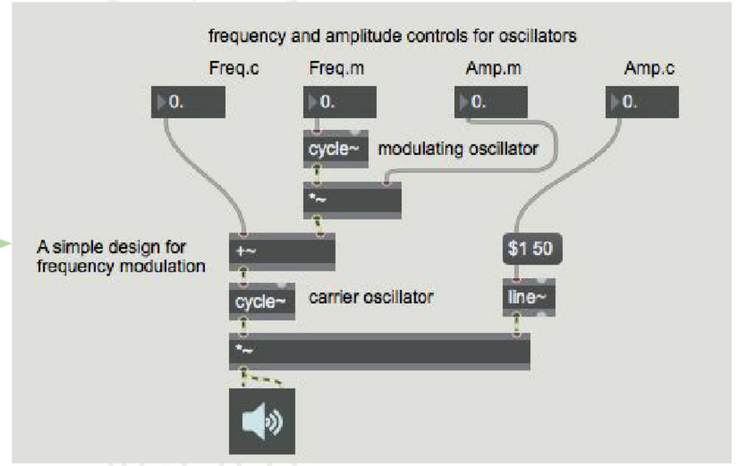


[0.5, 0.7]
[0.1, 0.1]
[0.5, 0.6]
[0.3, 0.9]
[0.4, 0]
[0.2, 0.7]
[0.5, 0.6]
[0.6, 0.1]
[0.5, 0.3]
[0.5, 0.7]
[0.1, 0.8]
[0.5, 0.1]
[0.4, 0.2]
[0.1, 0.8]
[0.5, 0.6]
[0.3, 0.9]
[0.4, 0.3]
[0.2, 0.7]

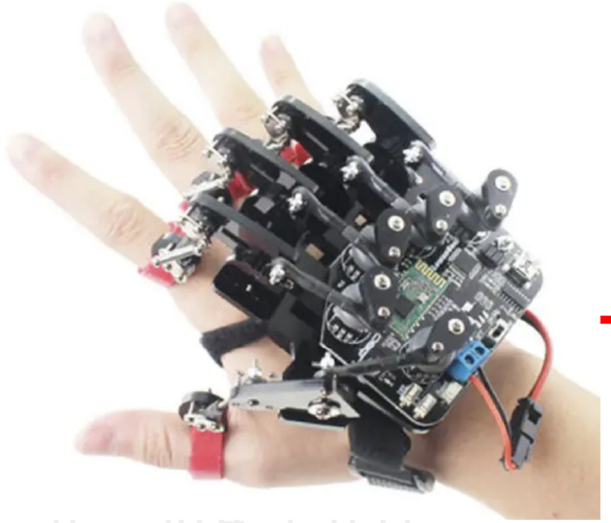
Neural Network *Predicting with Regression*

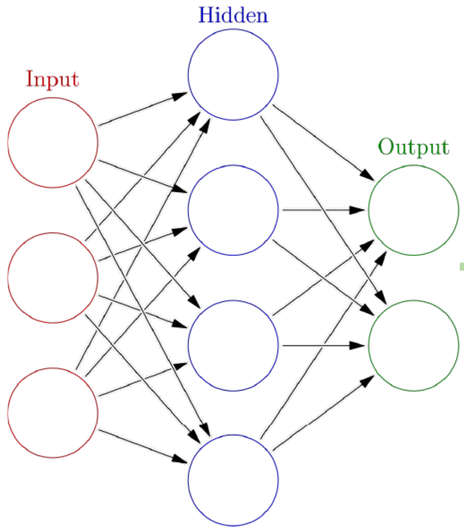


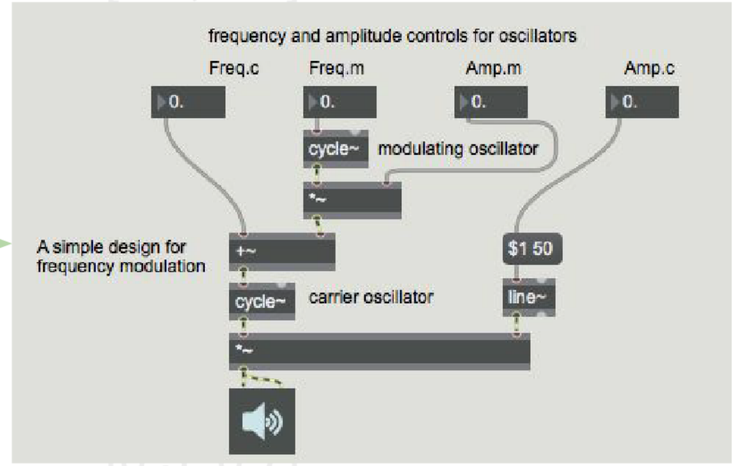




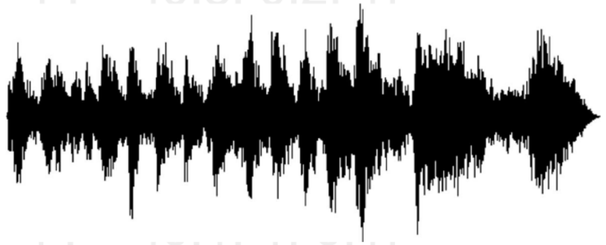
Neural Network *Predicting with Regression*



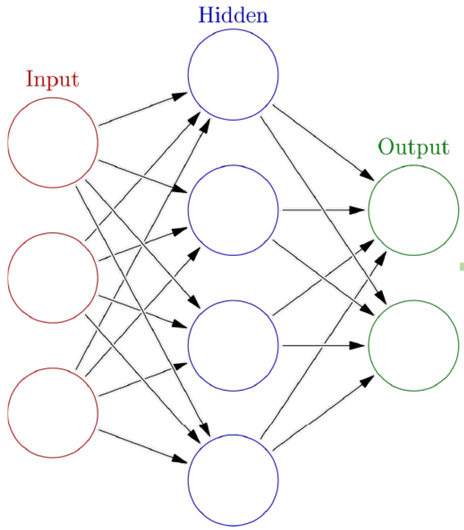


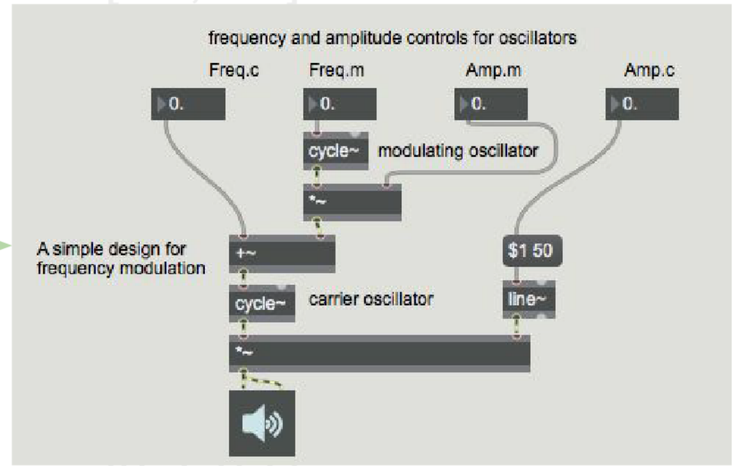


Neural Network *Predicting with Regression*

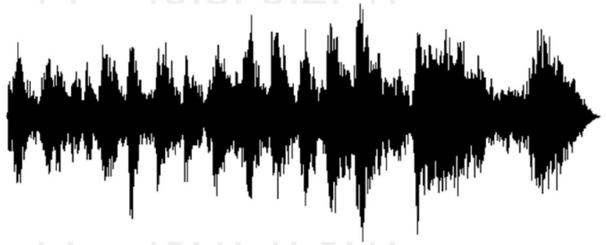


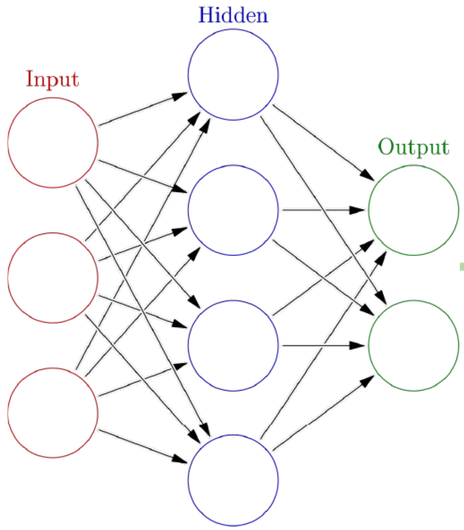
e.g.:
pitch
loudness
spectral centroid
spectral flatness





Neural Network *Predicting with Regression*

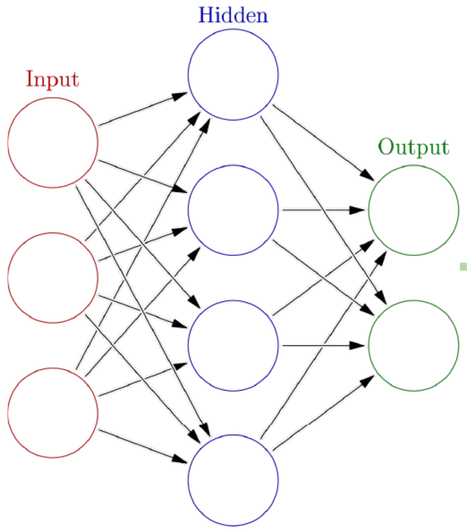




Neural Network *Predicting with Regression*

“a”: [0.1, 0.5, 1]
“b”: [0, 1, 1]
“c”: [1, 0, 1]
“d”: [0.2, 0, 1]
“e”: [0, 0, 0.5]
“f”: [0.8, 0.2, 1]
“g”: [0.9, 0.3, 0]
“h”: [0.4, 1, 0.7]
“i”: [0.2, 0.3, 1]
“j”: [0.1, 1, 0.1]
“k”: [0, 1, 1]
“l”: [0, 1, 1]
“m”: [0, 0, 1]
“n”: [0.4, 1, 0.7]
“o”: [0.2, 0.3, 1]
“p”: [0.1, 1, 0.1]
“q”: [0, 1, 1]
“r”: [0, 1, 1]

?



?

[0.5, 0.7]
[0.1, 0.1]
[0.5, 0.6]
[0.3, 0.9]
[0.4, 0]
[0.2, 0.7]
[0.5, 0.6]
[0.5, 0.7]
[0.1, 0.1]
[0.5, 0.6]
[0.2, 0.7]
[0.6, 0.1]
[0.5, 0.3]
[0.5, 0.7]
[0.1, 0.8]
[0.5, 0.1]
[0.4, 0.2]
[0.1, 0.8]

There are certain things that we care about, as musicians for example, that are really hard to articulate in code. It's hard for me to talk about what kind of quality of sound I want and then translate that into a set of filter coefficients. It's hard for me to talk about how I want a performer to move on stage and then translate that into some sort of mathematical equation for their trajectory. But it's a lot easier for me to either find examples of sounds that have a particular quality or to give examples of movements or if I'm using other types of modalities, often curating or creating examples are just way easier for us as people. And this relates to the types of tacit knowledge and embodied knowledge we bring to creative practices.

-Rebecca Fiebrink

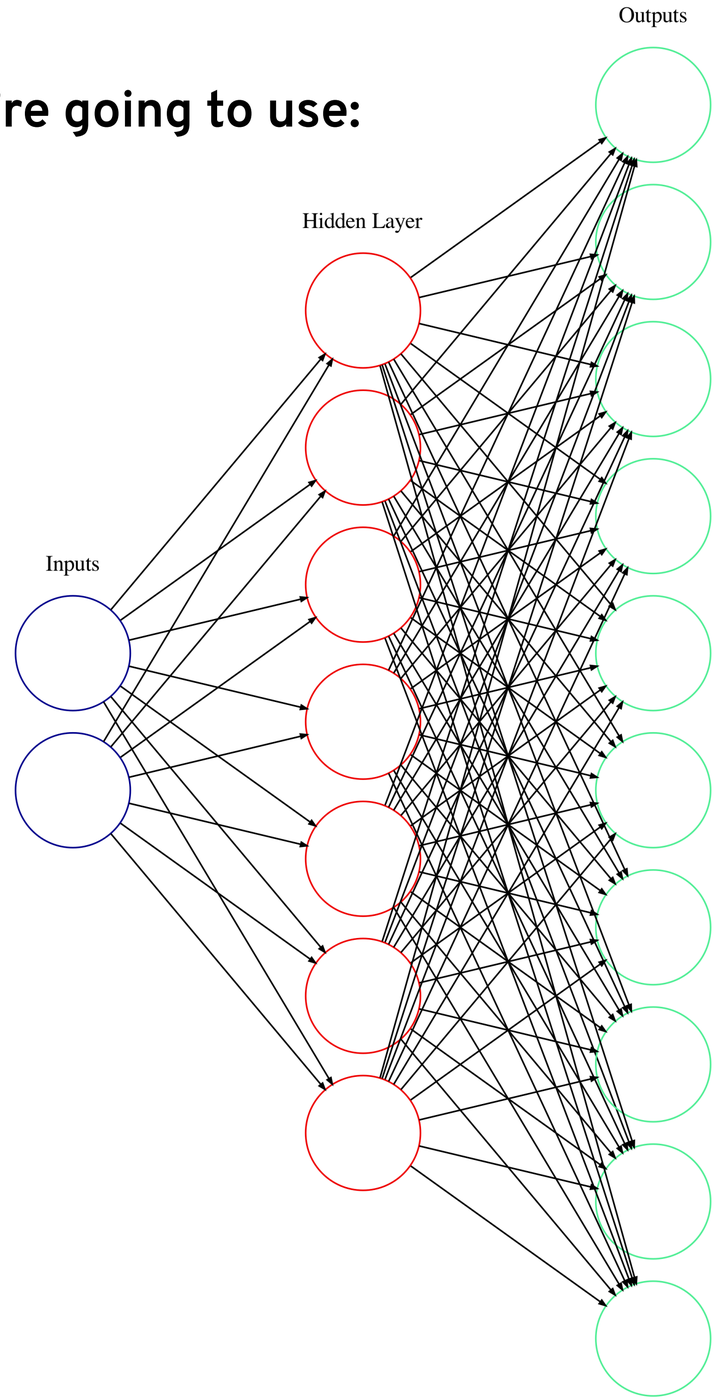
CeReNeM. 2019. Rebecca Fiebrink: Machine Learning as Creative Design Tool. Centre for Research in New Music. YouTube. Visited on 11/23/2020. https://www.youtube.com/watch?v=Qo6n8MuEgdQ.

Structure of the neural network we're going to use:

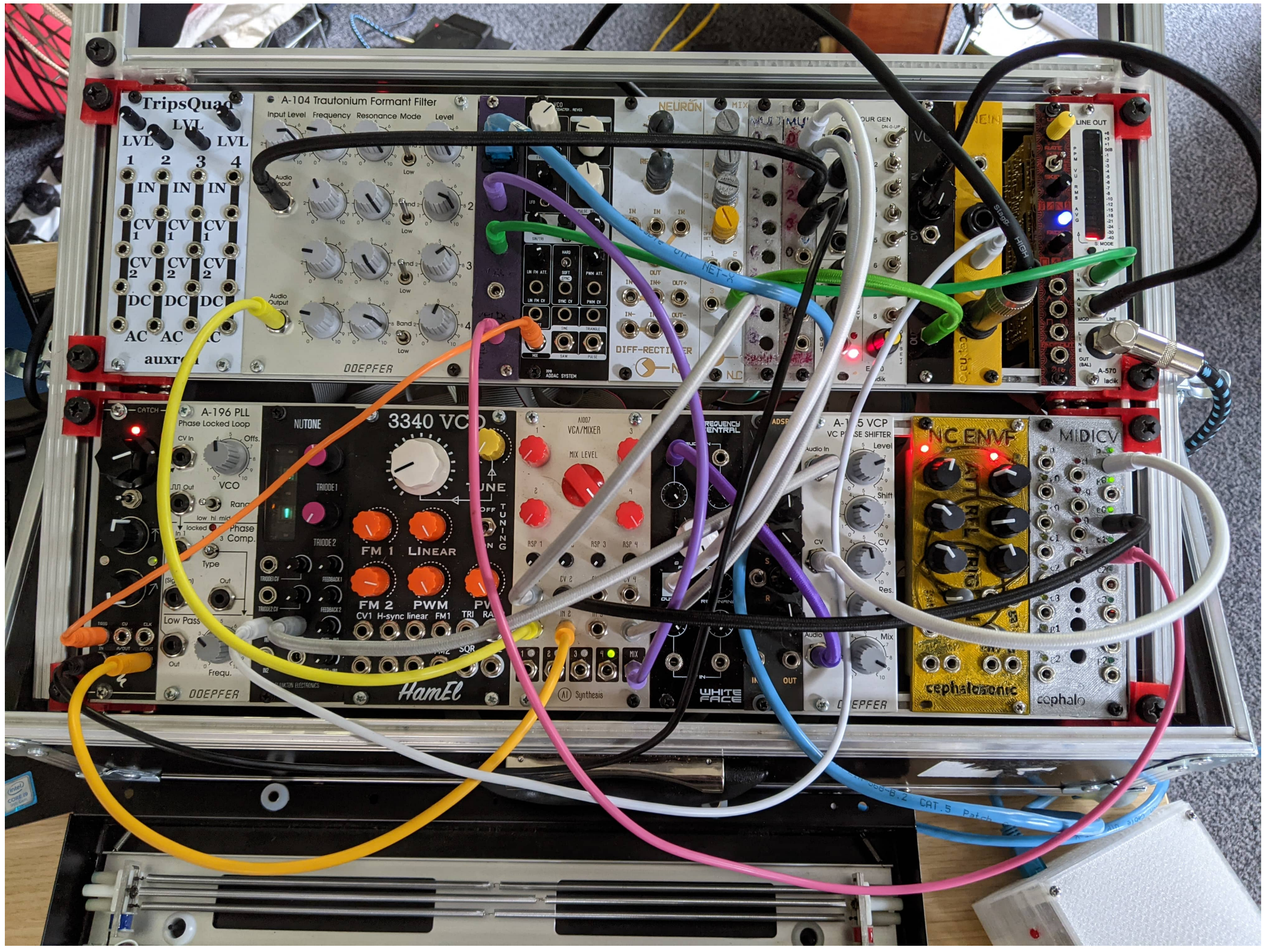
2 inputs

1 hidden layer of 7 nodes

10 outputs



fluid.mlpregressor~



FeedbackFeedforward
by Alice Eldridge and Chris Kiefer

learn.flucoma.org/explore/eldridge-kiefer