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

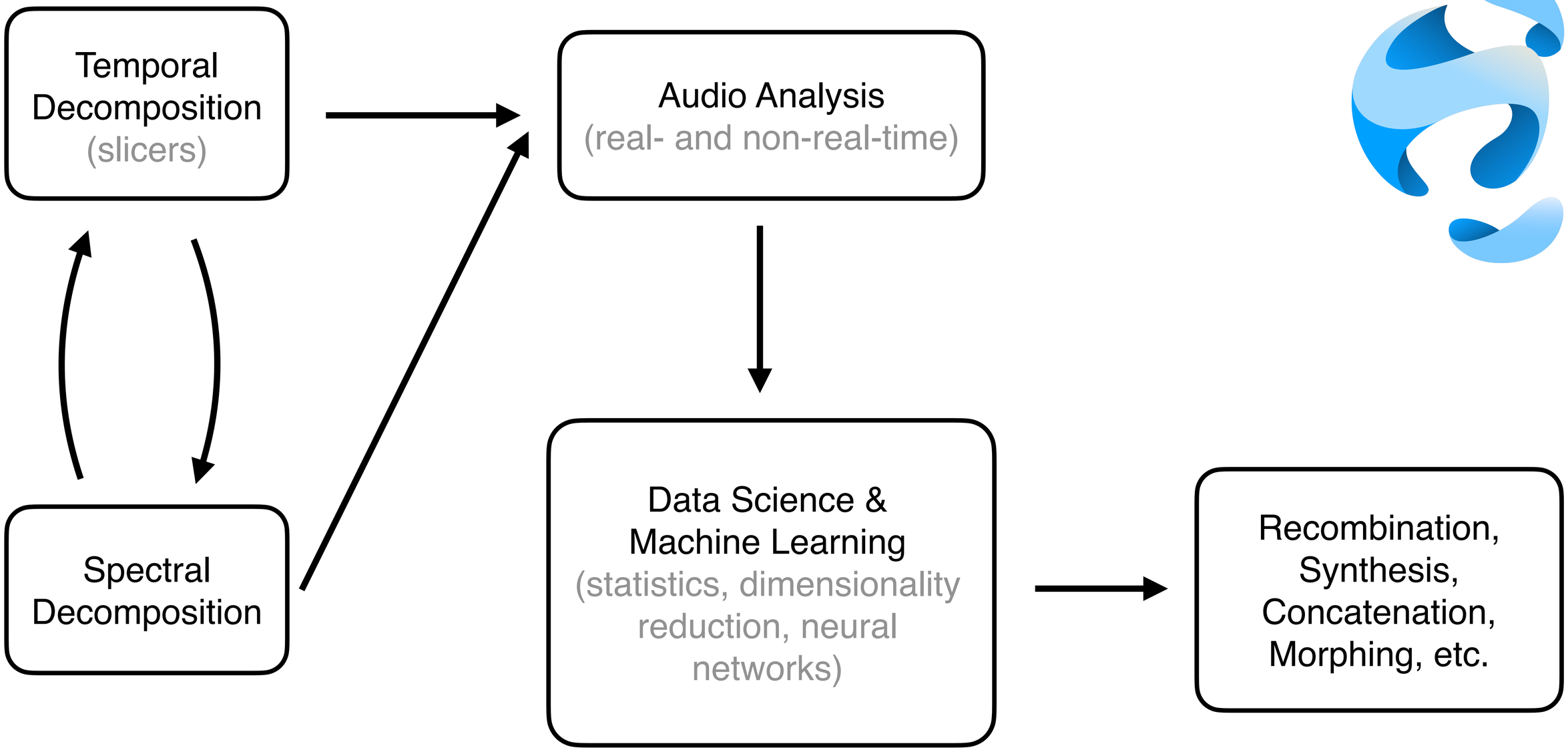
Decompose Audio
- extract transients
- harmonic/percussive separation
- model as sine waves
- non-negative matrix factorisation

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

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

Analyze 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.

Supervised Learning

Classification

- fluid.mlpclassifier~
- fluid.knnclassifier~

Regression

- fluid.mlpregressor~
- fluid.knnregressor~

Preprocessing

- fluid.normalize~
- fluid.standardize~
- fluid.robustscale~

Unsupervised Learning

Dimensionality Reduction

- fluid.pca~
- fluid.umap~
- fluid.mds~
- autoencoder (fluid.mlpregressor~)

Clustering

- fluid.kmeans~
- fluid.skmeans~

Linear Assignment

- fluid.grid~

Also…
fluid.bufnmf~

scikit-learn

Machine Learning in Python

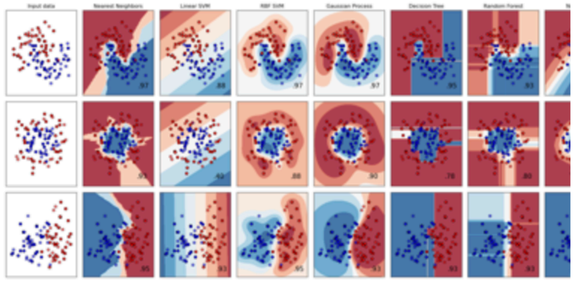
Getting Started | Release Highlights for 1.2 | GitHub

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

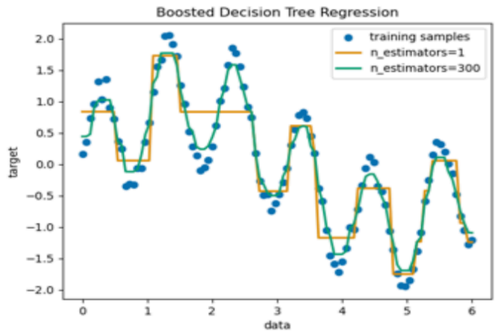


Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

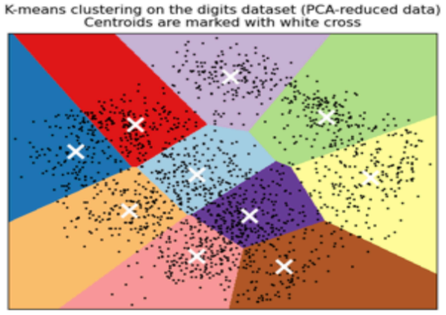


Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...



Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Model selection

Comparing, validating and choosing parameters and models.

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text

Supervised Learning

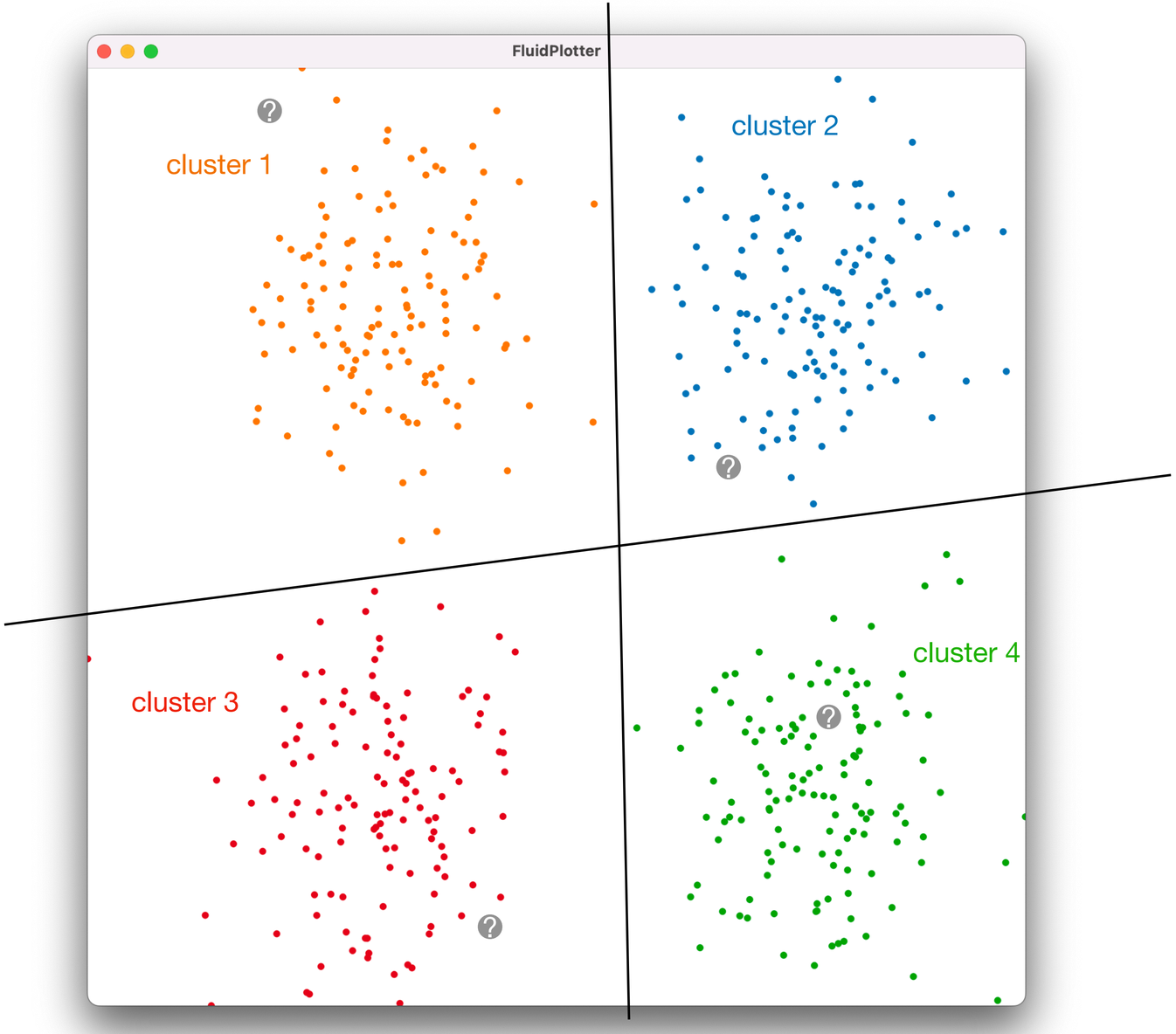
Learning patterns, associations, or relationships from data that is pre-labeled

Classification
Regression

Neural Networks
KNN

Unsupervised Learning

Learning/finding patterns and relationships in data that is not labeled



Dimensionality Reduction

Clustering

Feature Learning

Fluid Corpus Manipulation

More help / discussion at our forum

Check out the learn platform

Tutorials | Objects | Tasks | Inspiration | Examples

We've worked toward making a set of concise tutorials that cover a very wide range of objects and workflows found in the FluCoMa toolkit. We think that if you at least watch these three tutorials, you'll be familiar with most of the objects and be able to start building your own patches and systems out of the toolkit. Select a tutorial to explore below.

Select a tutorial...

- Building a 2D Corpus Explorer
- Classifying Sounds with a Neural Network
- Controlling a Synth with a Neural Network

Be aware, some of the patches you can open from within the package will differ slightly from the tutorial video's final result.

YouTube Video | Open Patch

Imagine you have a large corpus of sounds that you've collected from a studio session, some outside sound walks or experimenting with a synthesiser on a rainy afternoon. This tutorial teaches you how to build a '2D Corpus Explorer', a patch that will enable you to interrogate and listen to those sounds in a structured manner. The end result is similar to CataRT and AudioStellar in that small segments of your corpus sounds are analysed and mapped to a two-dimensional space that can be explored using your mouse. It will cover topics such as segmentation, analysis, data processing and visualisation.

Fluid Corpus Manipulation

More help / discussion at our forum

Check out the learn platform

Tutorials | Objects | Tasks | Inspiration | Examples

Select a category...

Slice Audio | Transform Audio

Analyse Audio | Analyse Data

Decompose Audio | Helpers and Utilites

If you have some corpus data, you'll probably want to analyse, manipulate and transform it. A variety of objects fall under this category and help you to make different manoeuvres around your data.

Containers

- fluid.dataset~
- fluid.labelset~

Statistical Analysis

- fluid.bufstats~
- fluid.stats

Scaling and Preprocessing

- fluid.normalize~
- fluid.standardize~
- fluid.robustscale~

Searching and Querying

- fluid.kdtree~
- fluid.datasetquery~

Unsupervised Machine Learning

- fluid.pca~
- fluid.kmeans~
- fluid.skmeans~
- fluid.mds~
- fluid.umap~
- fluid.grid~

Supervised Machine Learning

- fluid.knnregressor~
- fluid.knnclassifier~
- fluid.mlpregressor~
- fluid.mlpclassifier~

Fluid Corpus Manipulation

More help / discussion at our forum

Check out the learn platform

Tutorials | Objects | Tasks | Inspiration | Examples

A major part of the FluCoMa project involved commissioning composers to create works using the toolkit. These works are recorded and have also been investigated in detail by Jacob Hart (one of the team members) in "explore" articles. These long-form articles draw out how objects were used, and he has even created patches for you to plunder for your own creative endeavours. This is a great place to start if you're searching for musical inspiration with the FluCoMa Toolkit.

Select a composer(s)...

- Lauren Sarah Hayes
- Leafcutter John
- Olivier Pasquet
- Rodrigo Constanzo
- Alex Harker
- Sam Pluta
- Richard Devine
- Hans Tutschku
- Alice Eldridge & Chris Kiefer

YouTube Video | Learn Article

Lauren Sarah Hayes is a Scottish improviser and sound artist. Her music is a mix of experimental pop/live electronics/techno/noise/free improvisation and has been described as 'voracious' and 'exhilarating'. She is a sculptress of sound, manipulating, remixing, and bending voice, drum machines, analogue synths and self-built software live and physically. She is excited by what can happen in the vulnerable relationships between sound, space, and audience. Her shows are highly physical, making the performance of live electronic music more engaging for audiences. Over the last decade she has developed and honed a deliberately challenging and unpredictable performance system that explores the relationships between bodies, sound, environments, and technology.

Fluid Corpus Manipulation Toolkit

The FluCoMa toolkit to analyse, transform and learn from sounds

Description

The Fluid Corpus Manipulation toolkit provides an open-ended, loosely coupled set of objects for ex
learning.

Almost all objects for audio analysis or transformation have audio-rate and buffer-based versions, a

Many useful examples can be found in the help files as well as in the example folder, which is here:

```
File.realpath(FluidFilesPath("../../Examples")).openOS;
```

Contents

Slice Audio

on signals	on buffers	digest
FluidAmpGate	FluidBufAmpGate	Events from amplitude enevelope
FluidAmpSlice	FluidBufAmpSlice	Onsets from amplitude envelope
FluidOnsetSlice	FluidBufOnsetSlice	Spectral onset detector
FluidTransientSlice	FluidBufTransientSlice	Transient model onset detector
FluidNoveltySlice	FluidBufNoveltySlice	Novelty based onset detection on a choice of des

Analyse Audio

on signals	on buffers	digest
FluidPitch	FluidBufPitch	Choice of pitch descriptors
FluidLoudness	FluidBufLoudness	Loudness Descriptor

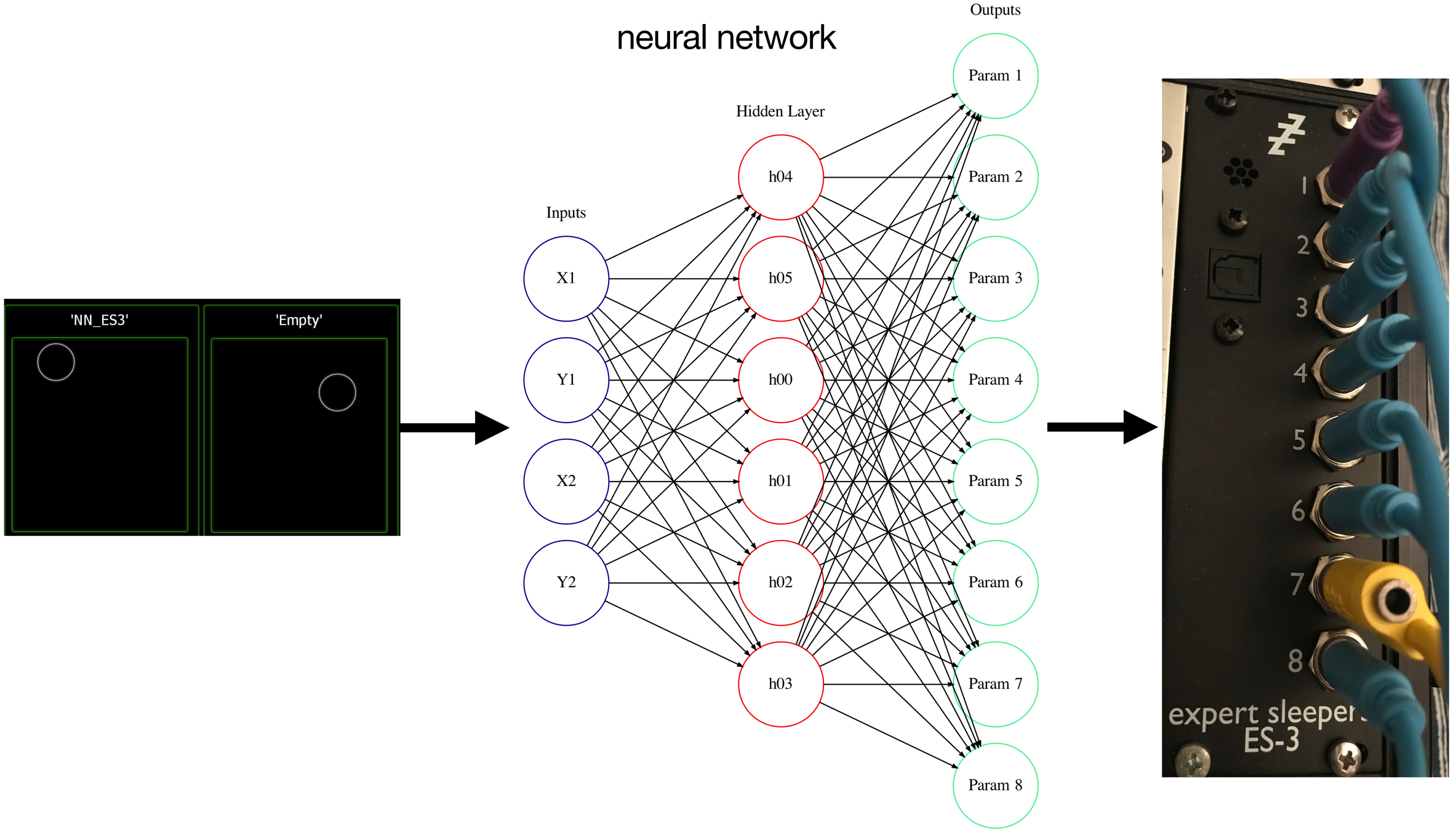
Plan for the Week…

MLPRegressor

control *many* synthesizer parameters from a smaller control space

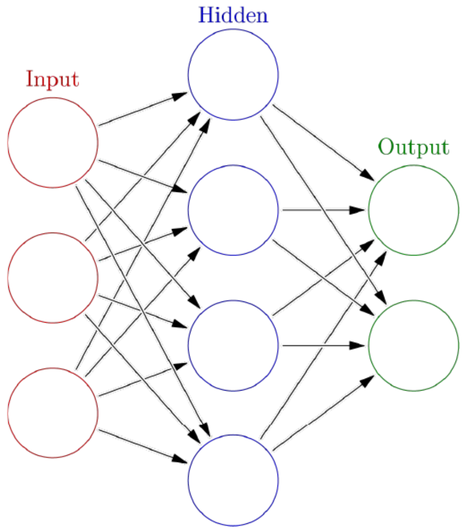
fluid.mlpregressor~


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

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*

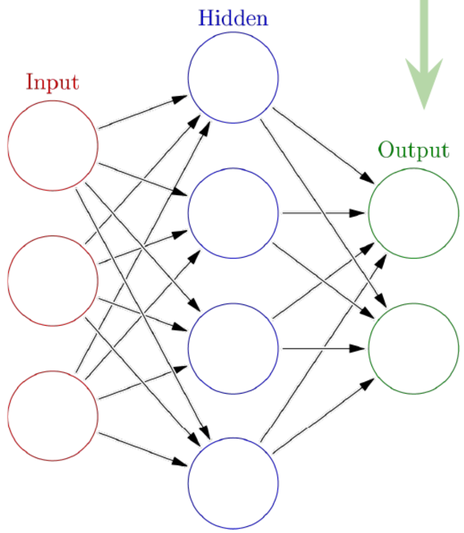
identifier

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.

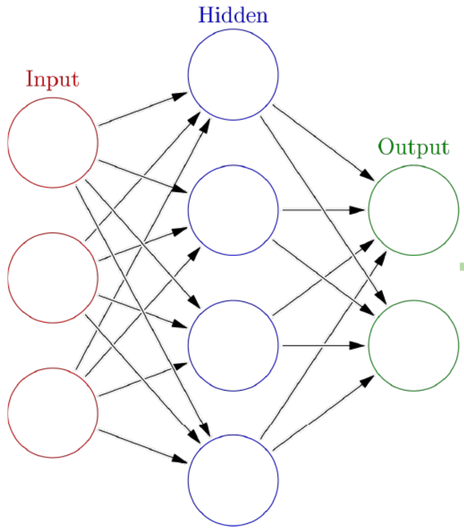
Neural Network *Predicting with Regression*

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]

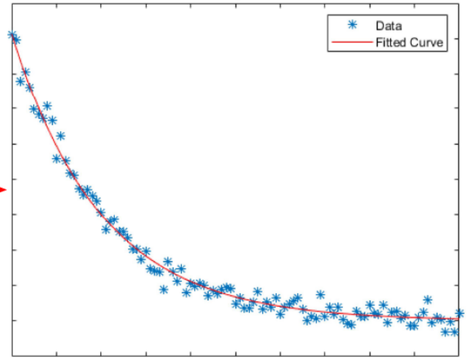
Neural Network *Predicting with Regression*

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]

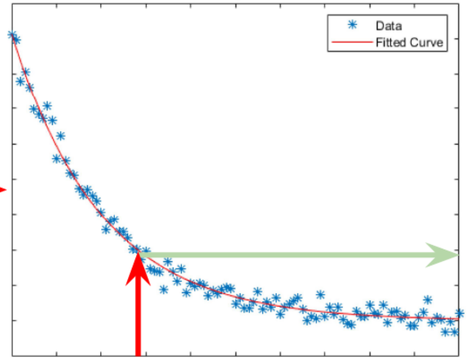
Neural Network *Predicting with Regression*

input examples

output examples

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

[0.6, 0.2, 0.4]

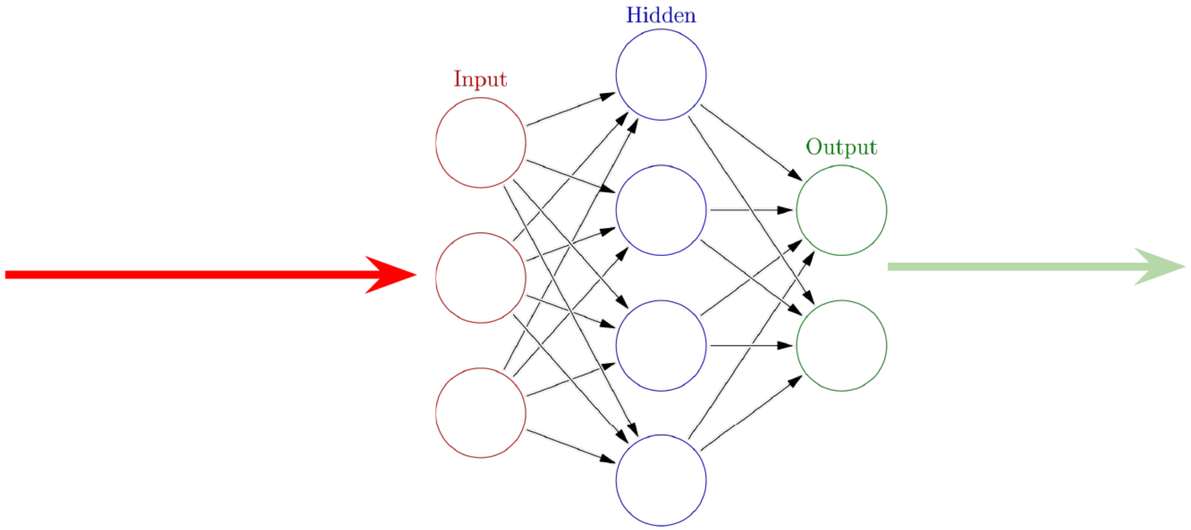


"here's an output that would 'fit' in your training data"

[0.9, 0.8]

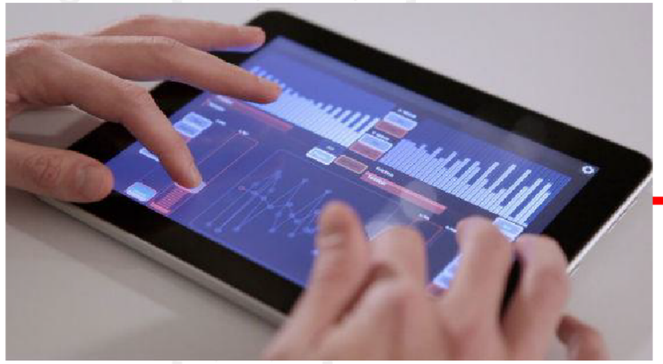
Neural Network *Predicting with Regression*

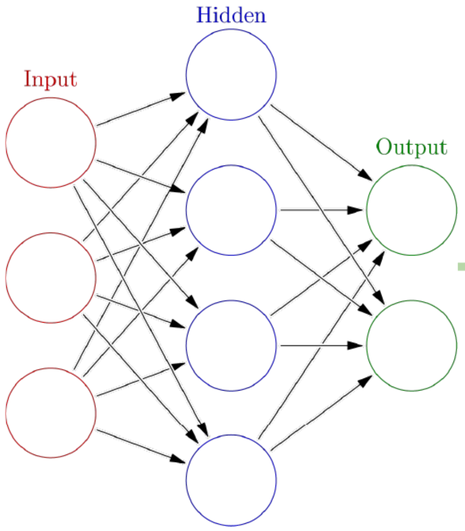
"a": [0.1, 0.5, 1]
"b": [0, 1, 1]
"c": [1, 0, 1]
"d": [0.2, 0, 1]
"e": [0, 0, 0.5]
"f": [0.8, 0.2, 1]
"g": [0.9, 0.3, 0]
"h": [0.4, 1, 0.7]
"i": [0.2, 0.3, 1]
"j": [0.1, 1, 0.1]
"k": [0, 1, 1]
"l": [0, 1, 1]
"m": [0, 0, 1]
"n": [0.4, 1, 0.7]
"o": [0.2, 0.3, 1]
"p": [0.1, 1, 0.1]
"q": [0, 1, 1]
"r": [0,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

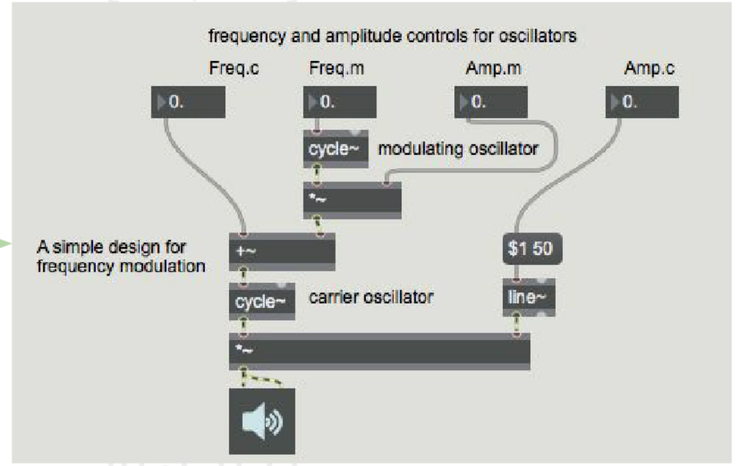


[0.5, 0.7]
[0.1, 0.1]
[0.5, 0.6]
[0.3, 0.9]
[0.4, 0]
[0.2, 0.7]
[0.5, 0.6]
[0.6, 0.1]
[0.5, 0.3]
[0.5, 0.7]
[0.1, 0.8]
[0.5, 0.1]
[0.4, 0.2]
[0.1, 0.8]
[0.5, 0.6]
[0.3, 0.9]
[0.4, 0.3]
[0.2, 0.7]

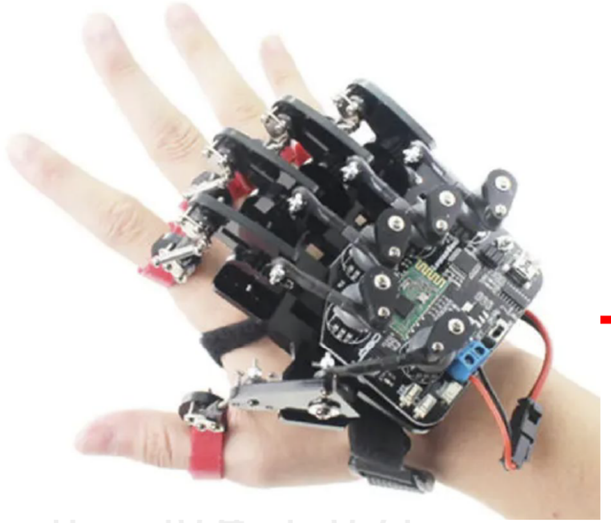
Neural Network *Predicting with Regression*

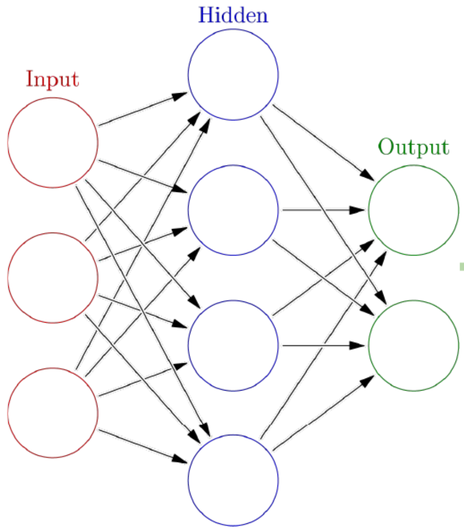


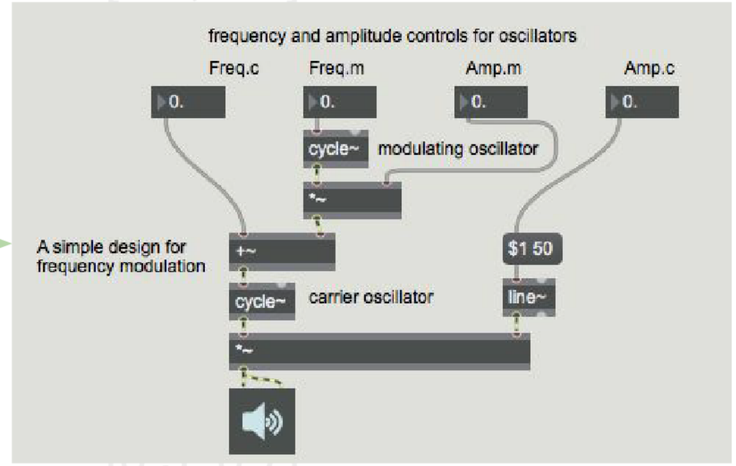




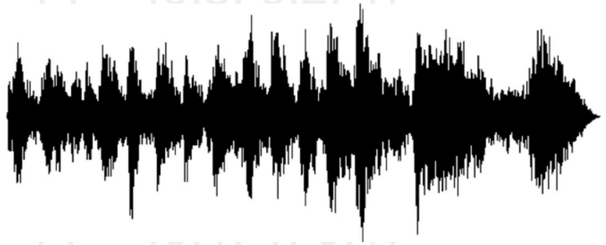
Neural Network *Predicting with Regression*



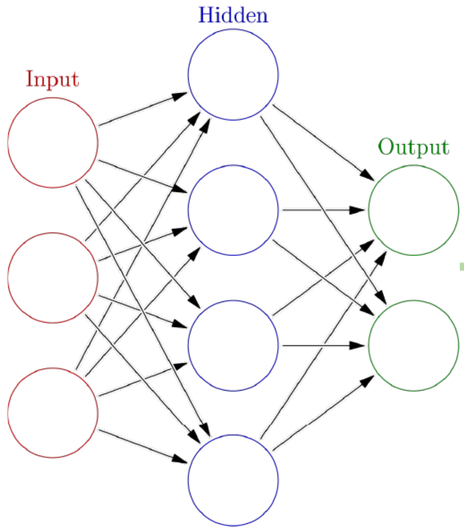


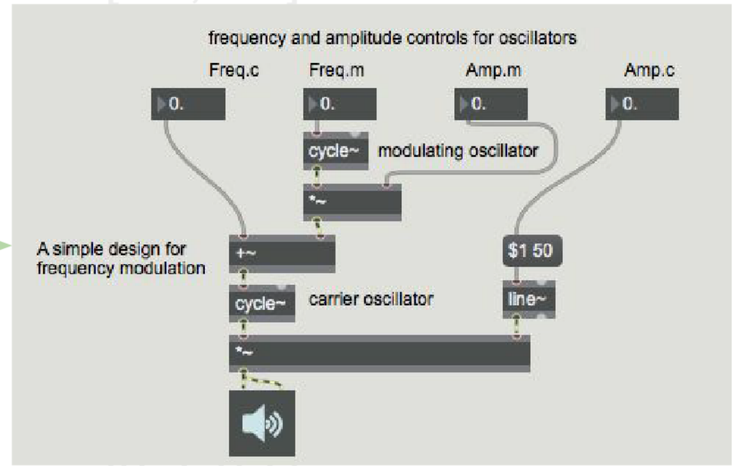


Neural Network *Predicting with Regression*

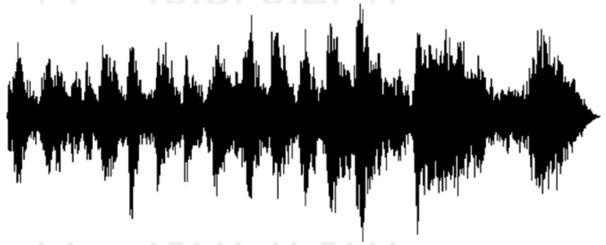


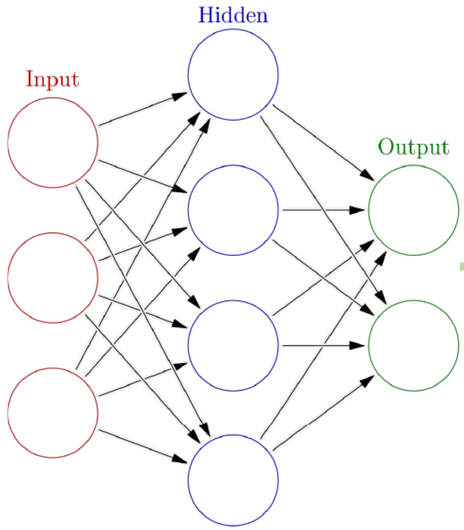
e.g.:
pitch
loudness
spectral centroid
spectral flatness





Neural Network *Predicting with Regression*

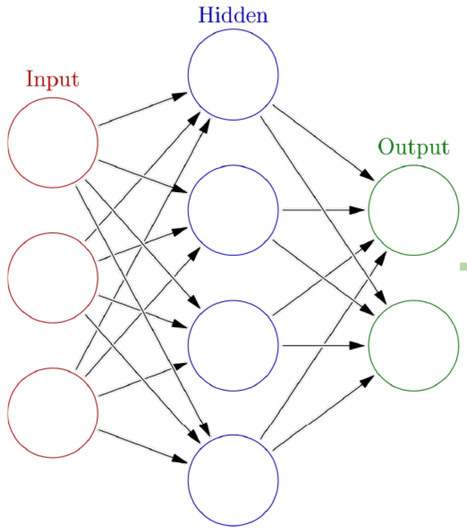




Neural Network *Predicting with Regression*

“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]

?



?

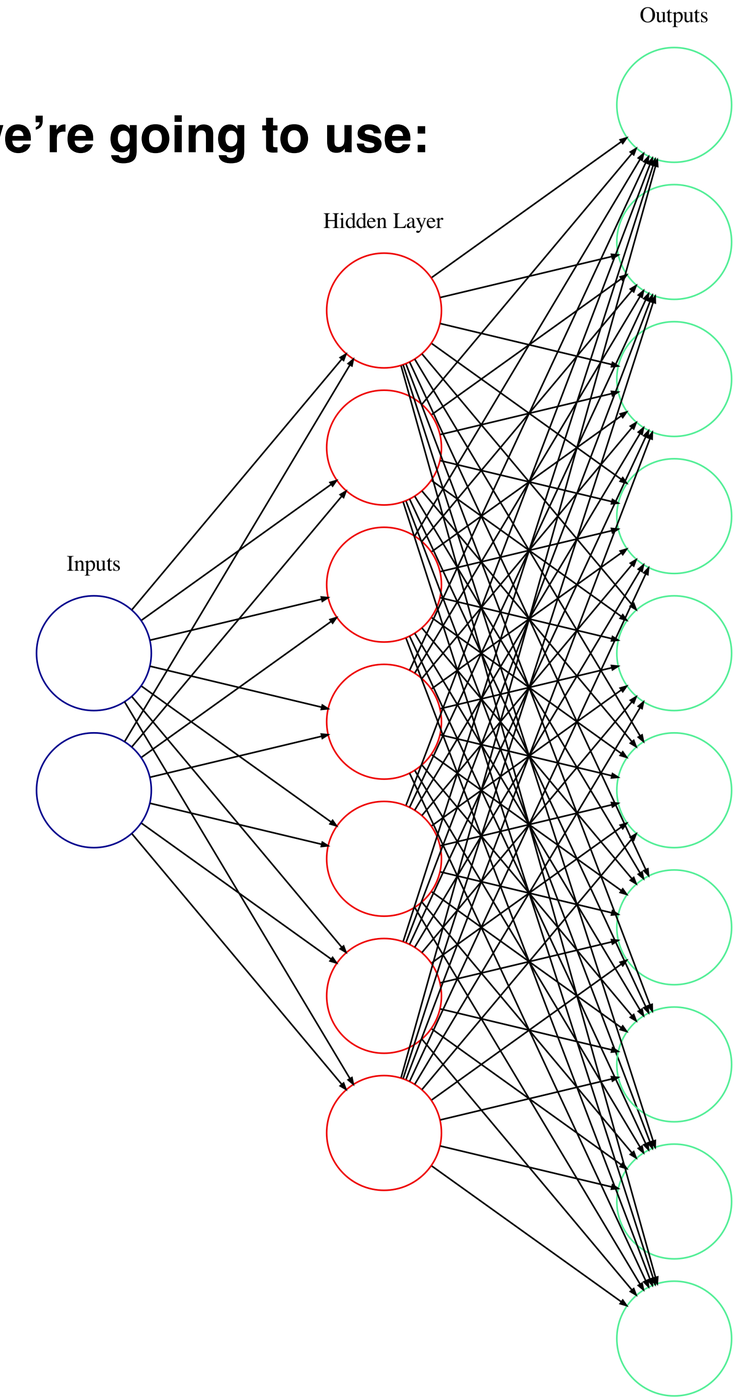
[0.5, 0.7]
[0.1, 0.1]
[0.5, 0.6]
[0.3, 0.9]
[0.4, 0]
[0.2, 0.7]
[0.5, 0.6]
[0.5, 0.7]
[0.1, 0.1]
[0.5, 0.6]
[0.2, 0.7]
[0.6, 0.1]
[0.5, 0.3]
[0.5, 0.7]
[0.1, 0.8]
[0.5, 0.1]
[0.4, 0.2]
[0.1, 0.8]

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

2 inputs

1 hidden layer of 7 nodes

10 outputs

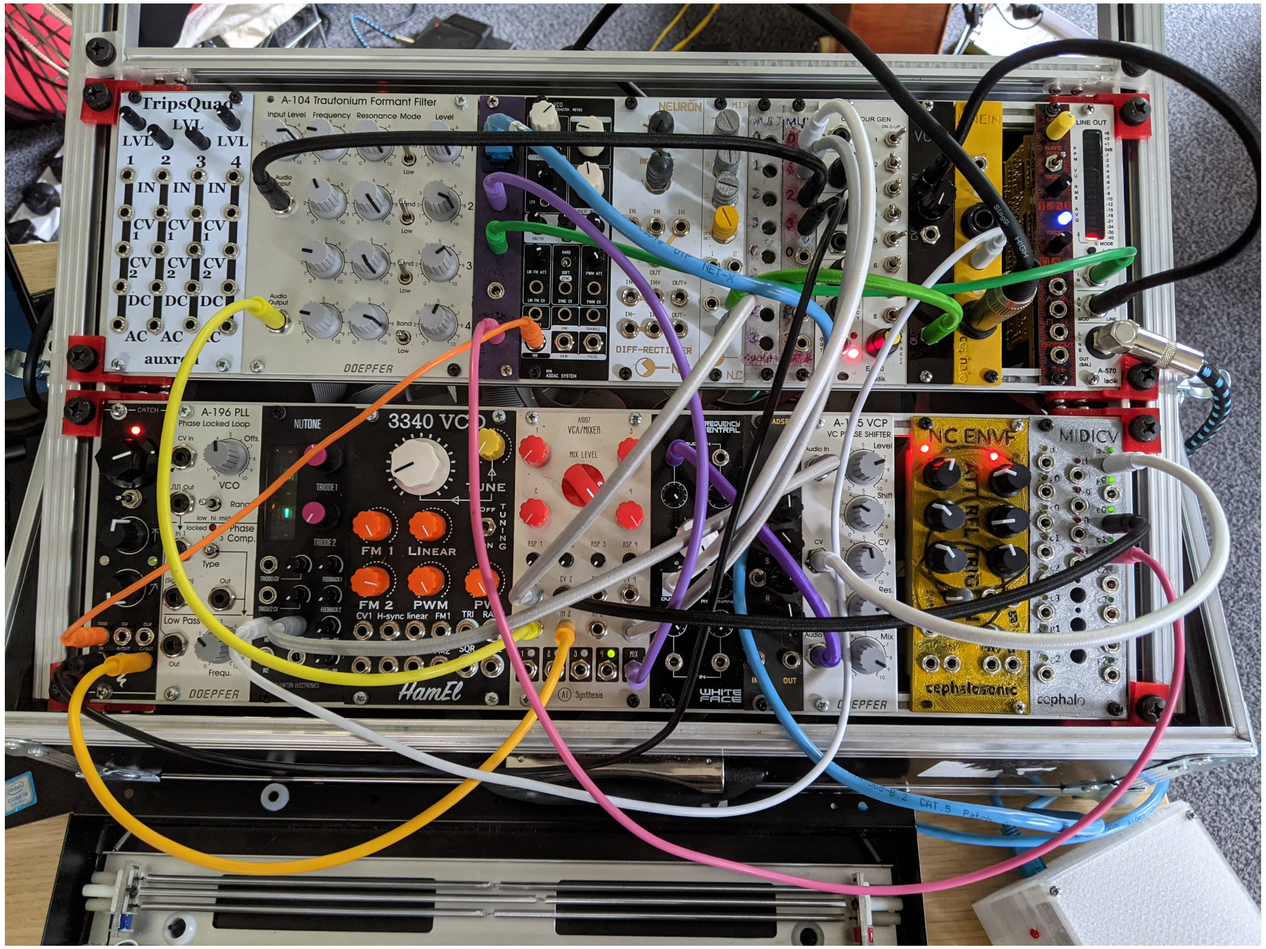


fluid.mlpregressor~

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.



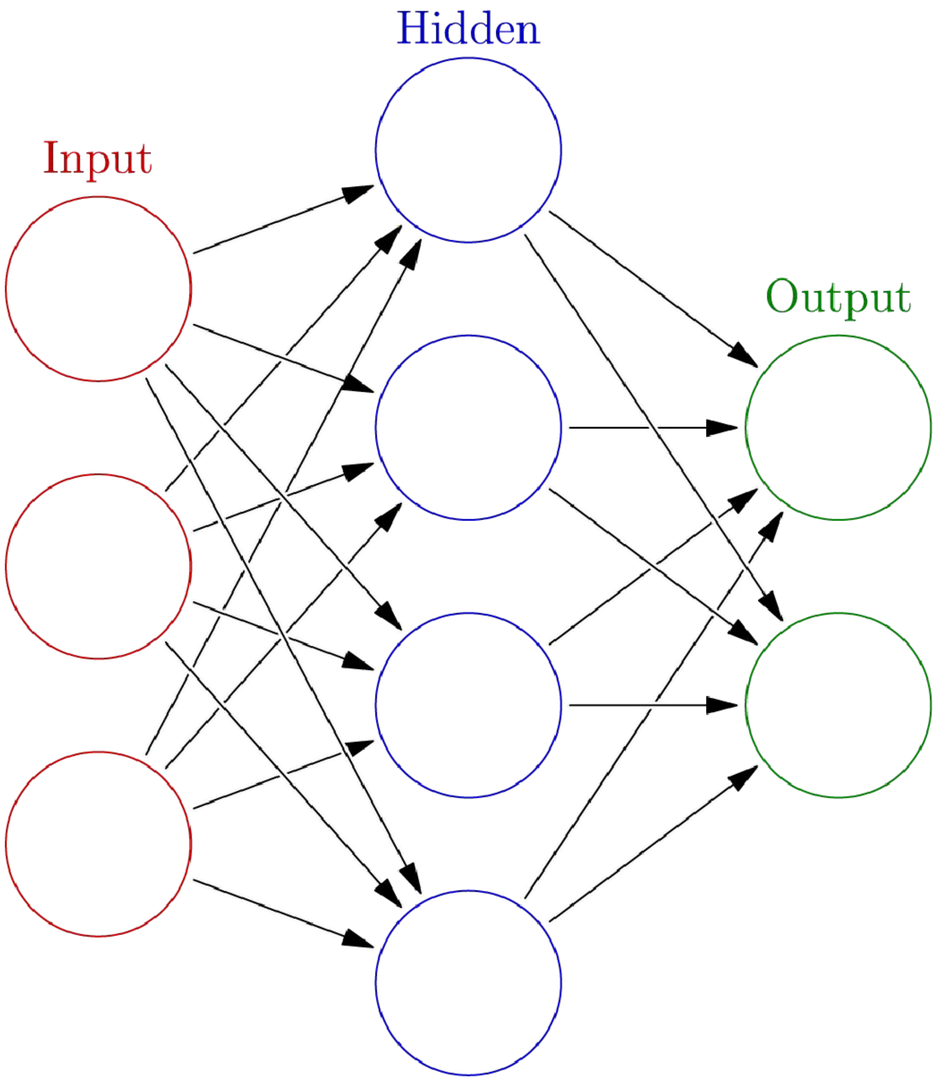
FeedbackFeedforward
by Alice Eldridge and Chris Kiefer

learn.flucoma.org/explore/eldridge-kiefer

FluidMLPClassifier

classify sounds by timbre in real-time

Neural Network (Multi-Layer Perceptron)

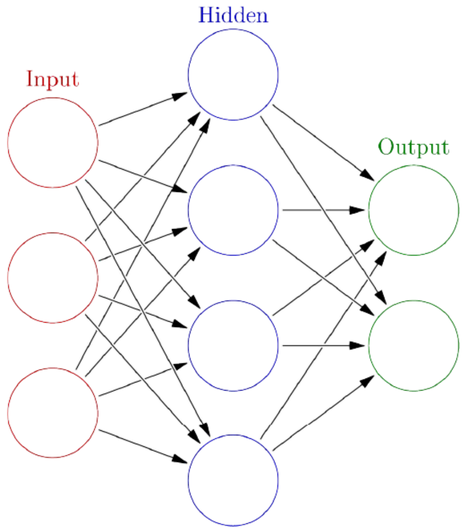


<u>Classification:</u>
a neural network predicts

which category (or “class”) an input belongs to

Neural Network *Training a Classifier*

trombone examples	oboe examples



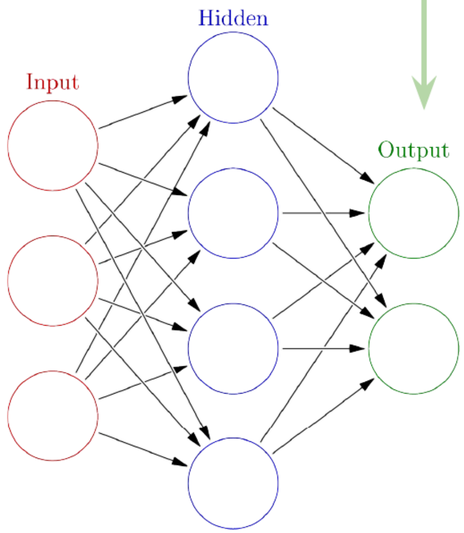
Neural Network *Training a Classifier*

trombone examples

oboe examples

"this example is a trombone"

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



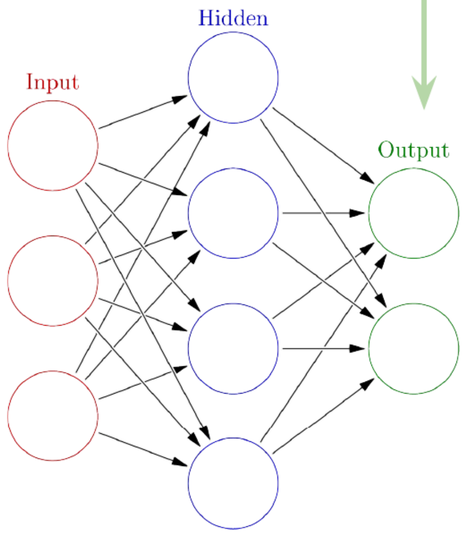
Neural Network *Training a Classifier*

trombone examples

oboe examples

"this example is an oboe"

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



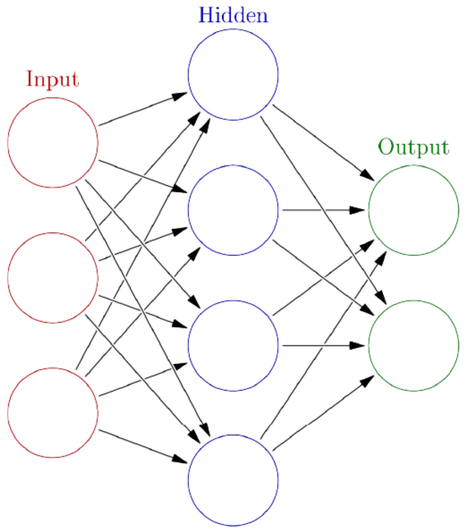
Neural Network *Predicting a Classification*

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”

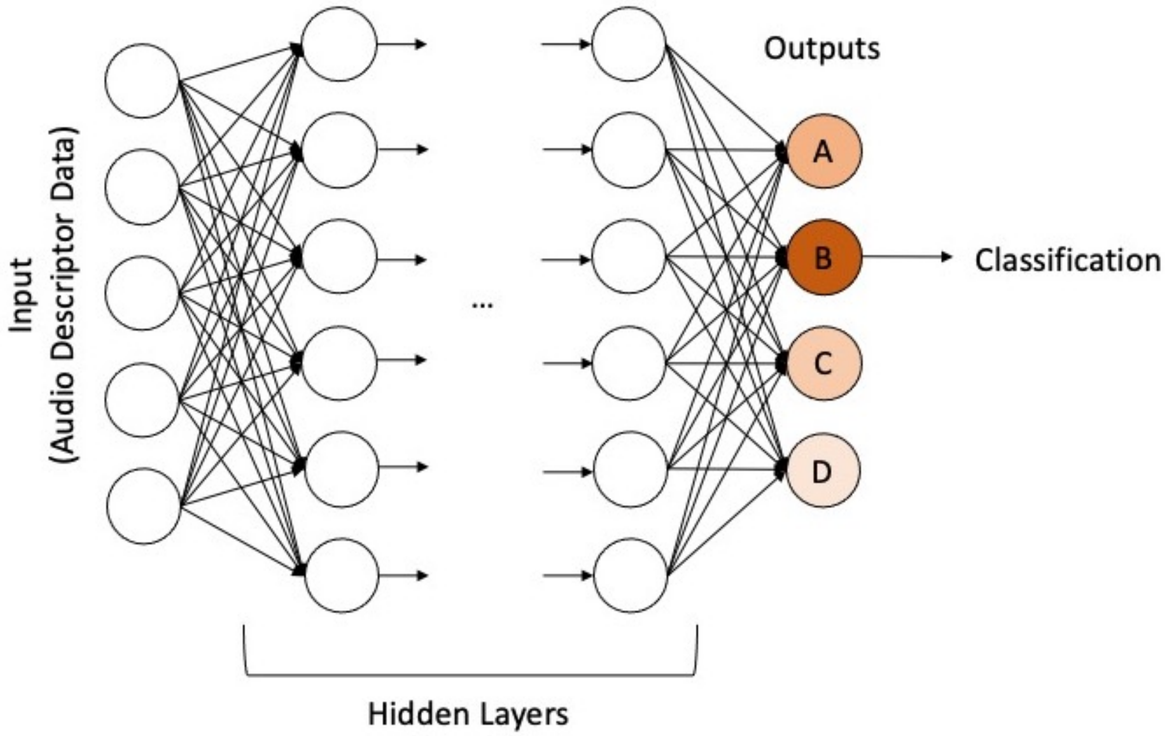
(or trombone…)

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