Online Workshop: "Performing with a virtual agent: machine learning for live coding"

Day 2

Anna Xambó

Music, Technology and Innovation - Institute for Sonic Creativity (MTI^2)
De Montfort University

Leicester Hackspace 27.1.2021

MIRLCAuto: A Virtual Agent for Music Information Retrieval in Live Coding

Partners: IKLECTIK, Leicester Hackspace, L'Ull Cec, Phonos, MTI²

Collaborators: TOPLAP Barcelona, FluCoMa, Freesound

Awarded with an EPSRC HDI Network Plus Grant

Online Workshop
Performing with a virtual agent: machine learning for live coding

London (IKLECTIK)
7/9/11.12.2020 - 19:00-21:00 (GMT)

Barcelona (L'Ull Cec)
11/13/15.1.2021- 19:00-21:00 (CET)

Leicester (Leicester Hackspace)
25/27/29.1.2021 - 19:00-21.00 (GMT)

More info at:
mirlca.dmu.ac.uk/workshops

Partners

IKLECTIK [off-site]

Collaborators

Materials / Working Tools

- **SuperCollider Extensions**: http://tiny.cc/LHS-SC-extensions
- **Working doc**: https://pad.riseup.net/p/online-workshop-lhs-keep
- **Zoom / Zoom chat**
- **GitHub repo (tutorials, tickets)**: https://github.com/mirlca/leicesterhackspace-workshop
- **Tutorials**: http://tiny.cc/LHS-tutorials (or under the Github link)
- **Project website**: https://mirlca.dmu.ac.uk
- **Project website**: https://mirlca.dmu.ac.uk

Schedule

- **Recap (all) - 10'**
- **Project presentation 2nd part: task 1 (Anna) - 15'**
- **Tutorial FluidMLPClassifier (Anna) - 10'**
- **Tutorial MIRLCa performance (Anna) - 15'**
- **Comfort break - 10'**
- **Hands-on MIRLCa performance (breakout rooms) - 20'**
- **Tutorial MIRLCa training (Anna) - 15'**
- **Hands-on MIRLCa training (breakout rooms) - 15'**
- **Plenary and closing (all) - 10'**

An introduction to machine learning in live coding and task 1: Train an agent to assist meaningful querying to Freesound.org

Learning Outcomes

- Get a sense of the practice of live coding (music live performance using code) by manipulating online crowdsourced sounds and the automatic use of feature descriptors obtained from freesound.org.

- **Get insight on a participatory design approach to designing a prototype for live coding performance.**

- **Get familiar with the application of neural networks, in particular a multilayer perceptron used as a classifier, to improve the practice of live coding with crowdsourced sounds.**

- **Be exposed to the main steps to solve a problem using machine learning techniques: the creation of a dataset, training a model, testing the model, and performing with / evaluating the model in an iterative cycle.**

- Understand how to combine different technologies in SuperCollider to build a prototype for live coding performance.

At the end of the workshop you will be able to…

- Use SuperCollider and the MIRLC2 library to retrieve sounds from Freesound.org based on a live coding approach.
- **Use a trained model using the FluCoMa library to retrieve sounds that are based on personal musical taste.**
- **Train your own model using the FluCoMa library to retrieve sounds that are based on your personal musical taste.**
- Analyse how to define a virtual agent that can react to the live coder inputs using the FluCoMa library.
- Explore creative strategies to perform with a virtual agent using machine learning for live coding.

"Crowdsourced Eulerisms". Eulerroom Equinox 2020.
Streaming from Sheffield, UK. March 23, 2020.

MIRLCRep 2.0

Unwanted Situations: The Guitar Case

n02-peterMann

from noiselets by carpal tunnel

04:13 / 11:10

Digital Track

Streaming + Download

Includes high-quality download in MP3, FLAC and more. Paying supporters also get unlimited streaming via the free Bandcamp app.

Buy Digital Track name your price

Send as Gift

Buy the Full Digital Album

from noiselets, released January 8, 2018
List of sounds used from Freesound.org coming soon.

Share / Embed Wishlist

https://carpal-tunnel.bandcamp.com/track/n02-petermann (around 04:26)

Research Question

Can we build a **virtual agent live coder companion** that **learns** from human live coders using **machine learning** algorithms and a **large dataset of sounds** which goes beyond the approach of following live coder actions (also known as the call-response strategy) and creates **legible and negotiable actions**?

How?

1. Identify ML tasks.
2. Implement the ML tasks with suitable tools.
3. Test the implemented ML tasks.

1. Identify ML Tasks

Identify ML Tasks

- **Two tasks** identified (**supervised learning**):
  1. **NN-1** learns my musical taste when retrieving sounds from Freesound: do I like it or not?
  2. **NN-2** learns to reply (call-response) with another query based on the existing sound and my musical taste. The response can be based on pitch, bmp or similarity.
- **For each NN**:
  - Phase 1. **Training**.
  - Phase 2. **Testing**.

Identify ML Tasks

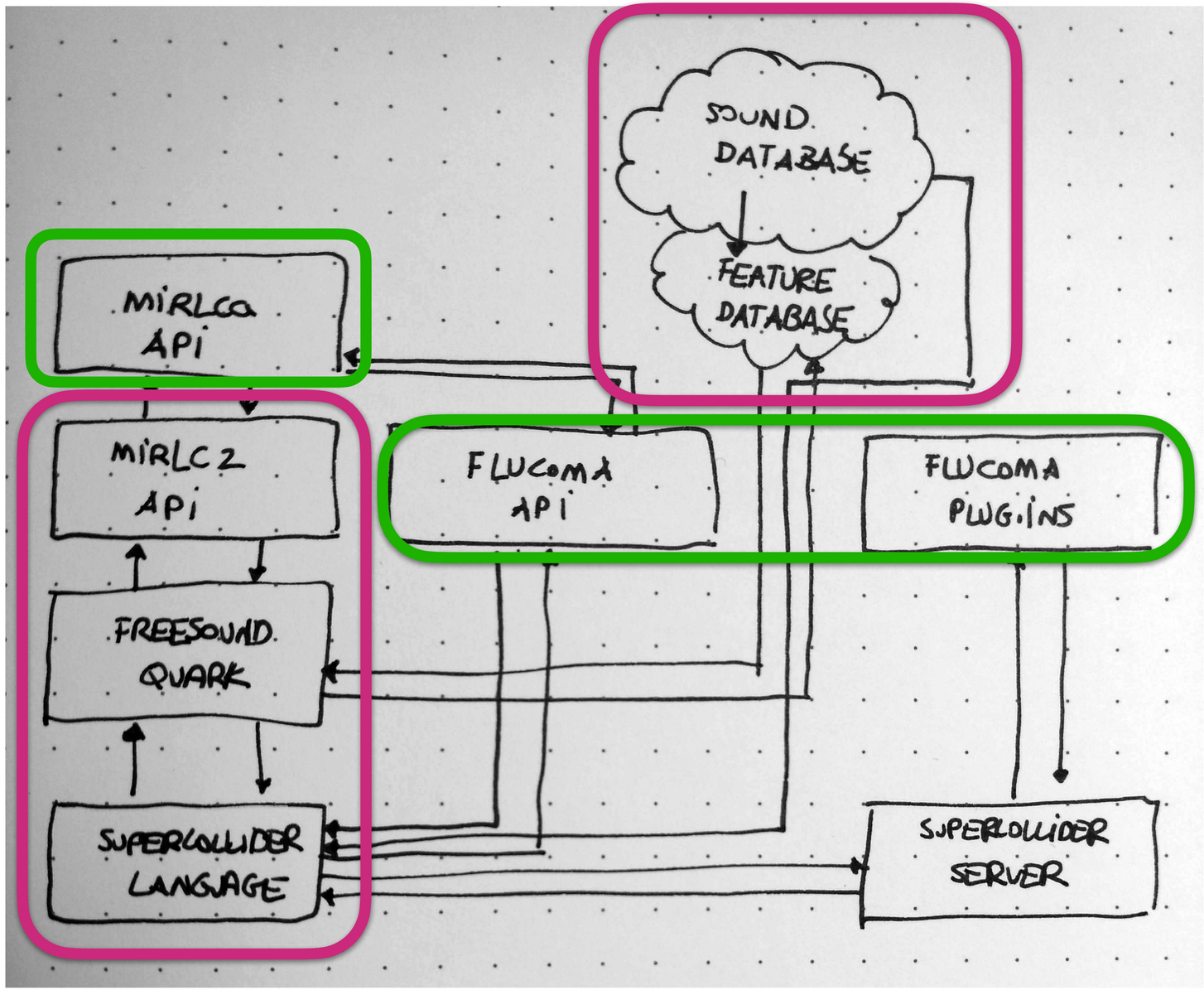
- **Two tasks** identified:
  1. **NN-1** learns my musical taste when retrieving sounds from Freesound: do I like it or not?
  2. **NN-2** learns to reply (call-response) with another query based on the existing sound and my musical taste. The response can be based on pitch, bmp or similarity.
- **For each NN**:
  - Phase 1. **Training**.
  - Phase 2. **Testing**.

NN-1: Learning my musical taste

1. Creation of a dataset.
2. Identifying suitable feature descriptors that describe the sound.
3. Training a suitable machine learning model => *this results in a file that you will try in the tutorial.*

https://mirlca.dmu.ac.uk/posts/towards-learning-my-musical-taste/

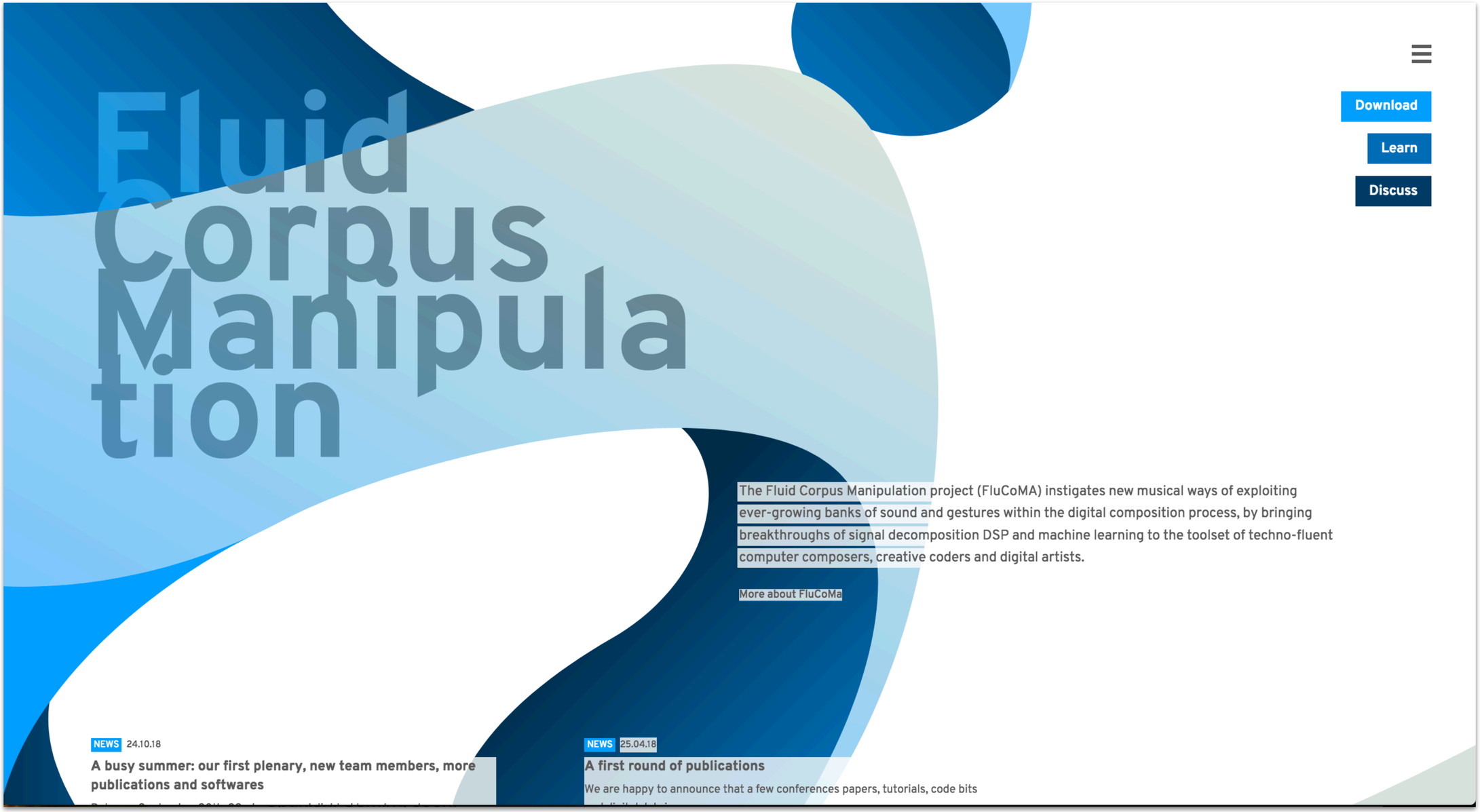
2. Implement the ML tasks with suitable tools



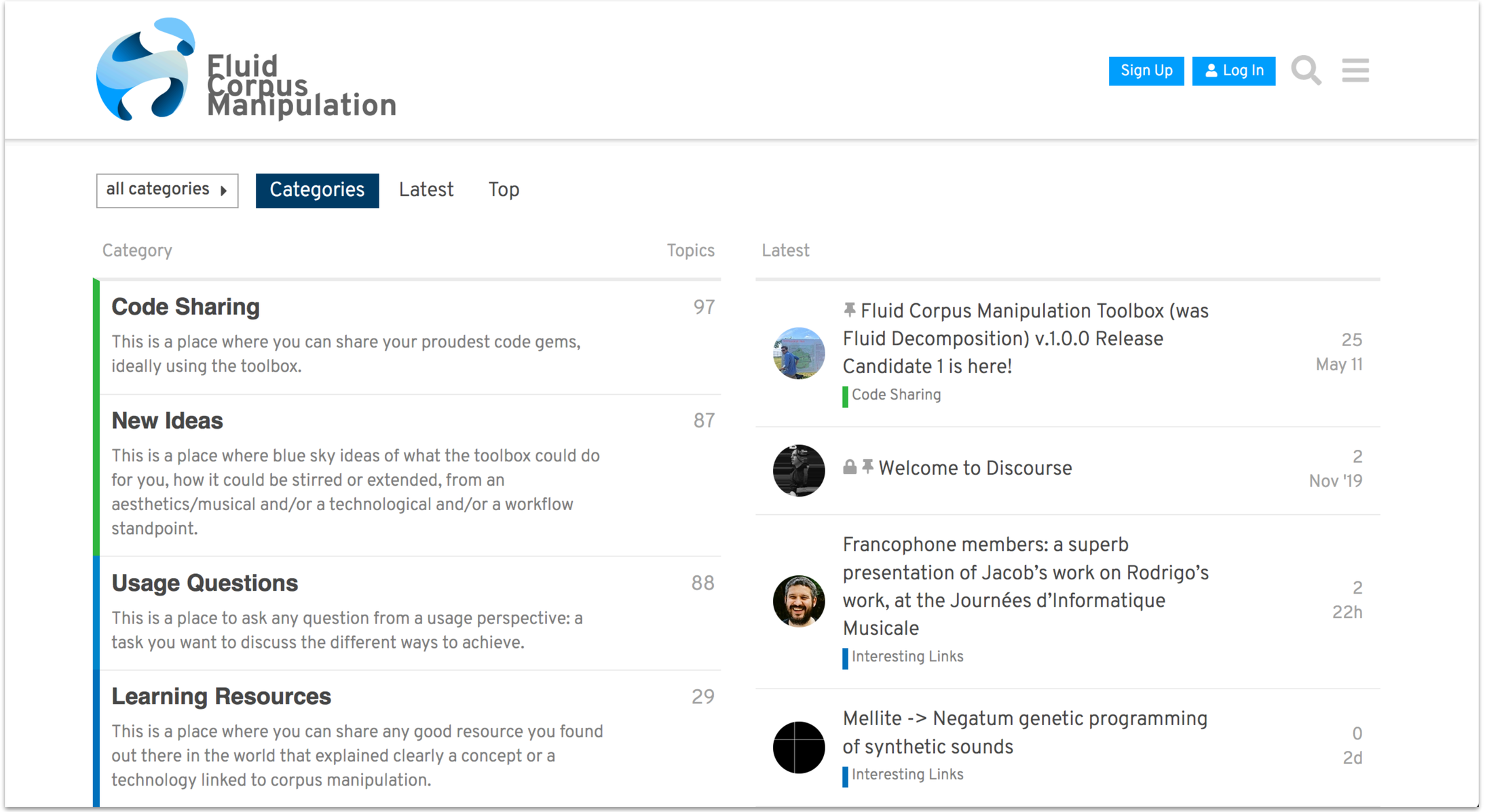
Day 1

Days 2 & 3

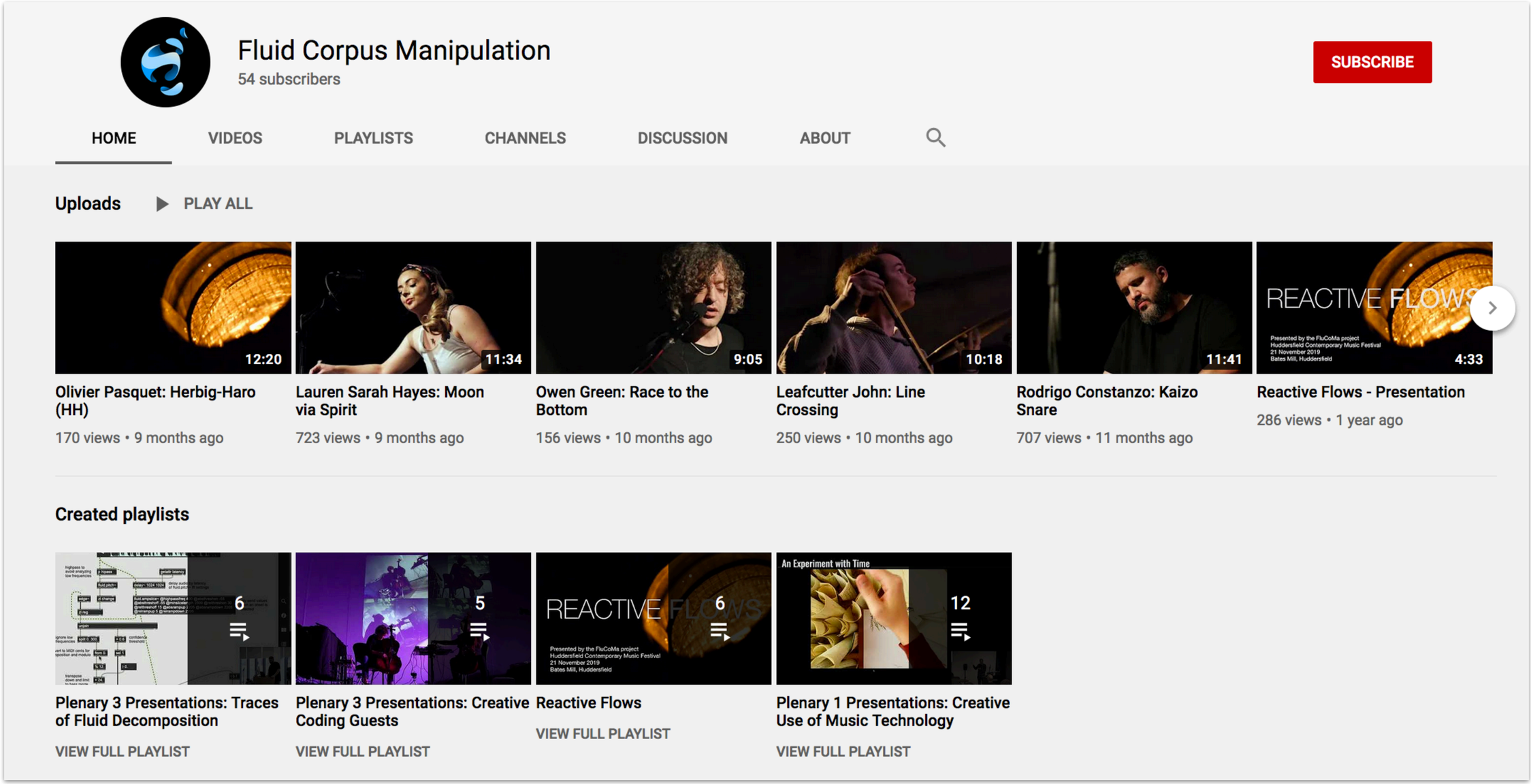
System's architecture



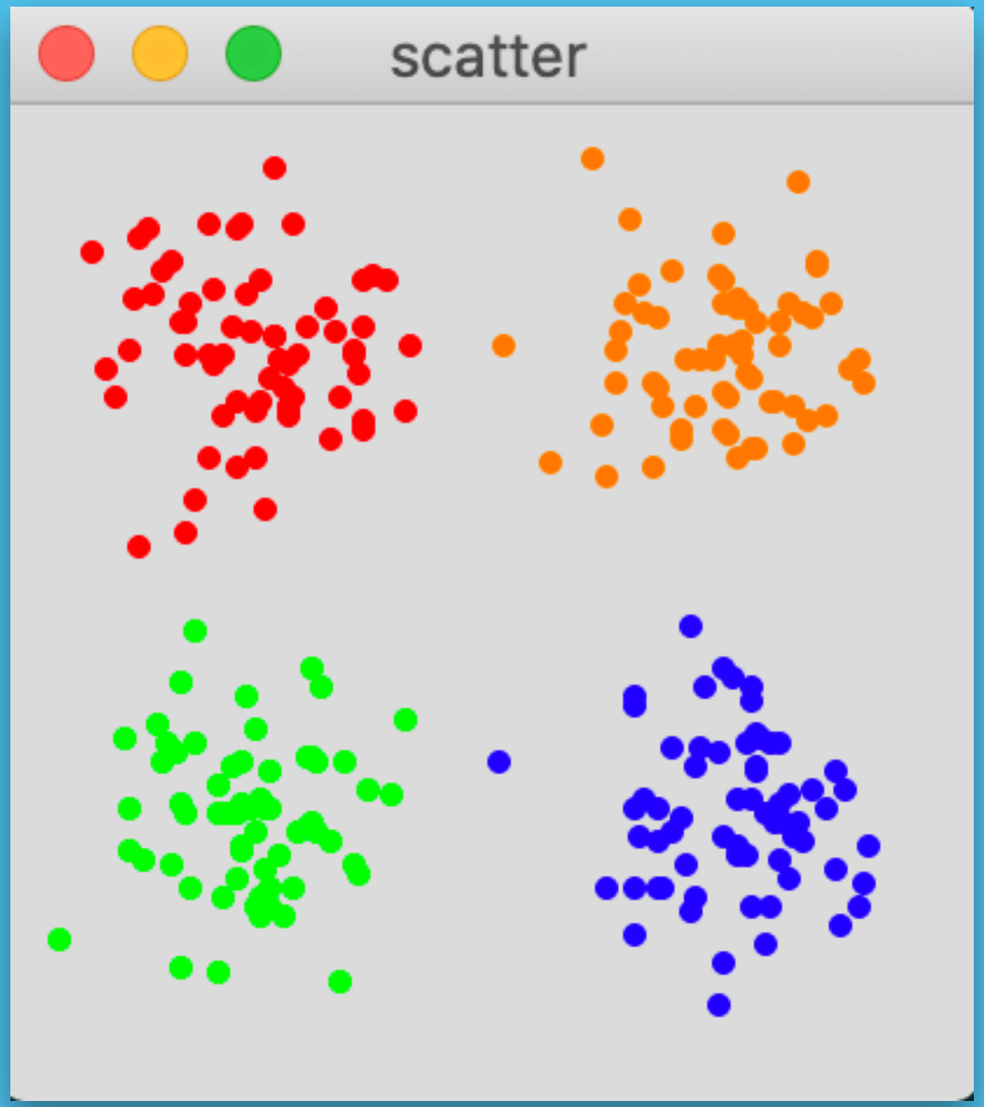
https://www.flucoma.org



https://discourse.flucoma.org



Fluid Corpus Manipulation YouTube channel:
https://www.youtube.com/channel/UCw44GjWHPQs0PKnj2OJmWCA



Flucoma Multilayer Perceptron (MLP) neural network (NN)

- **FluidMLPClassifier:** Perform classification using a MLP NN (we are using it for the NN-1 task).

www.flucoma.org

Classes (extension) | Machine learning

FluidMLPClassifier : FluidRTDataClient : FluidDataClient : FluidManipulationClient : Object

Extension

Classification with a multi-layer perceptron

Source: FluidMLP.sc

See also: FluidMLPRegressor, FluidDataSet

Perform classification between a FluidDataSet and a FluidLabelSet using a Multi-Layer Perception neural network.

Class Methods

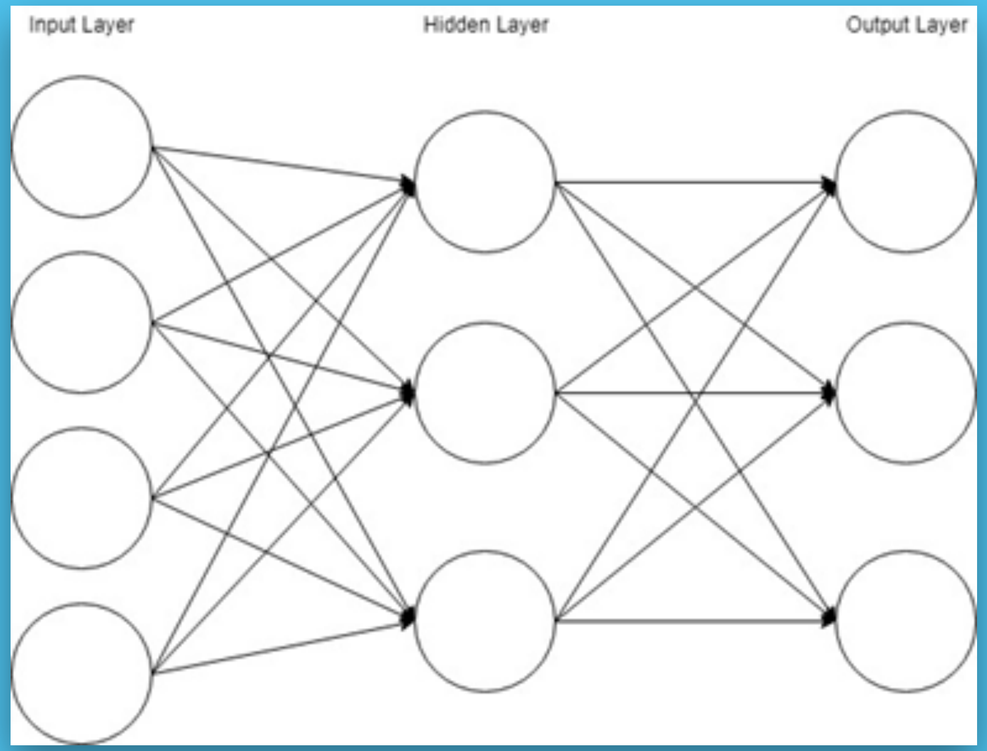
```
FluidMLPClassifier.new(server, hidden: [ 3, 3 ], activation: 2,
      maxIter: 1000, learnRate: 0.0001, momentum: 0.9, batchSize: 50,
      validation: 0.2)
```

Creates a new instance on the server.

Arguments:

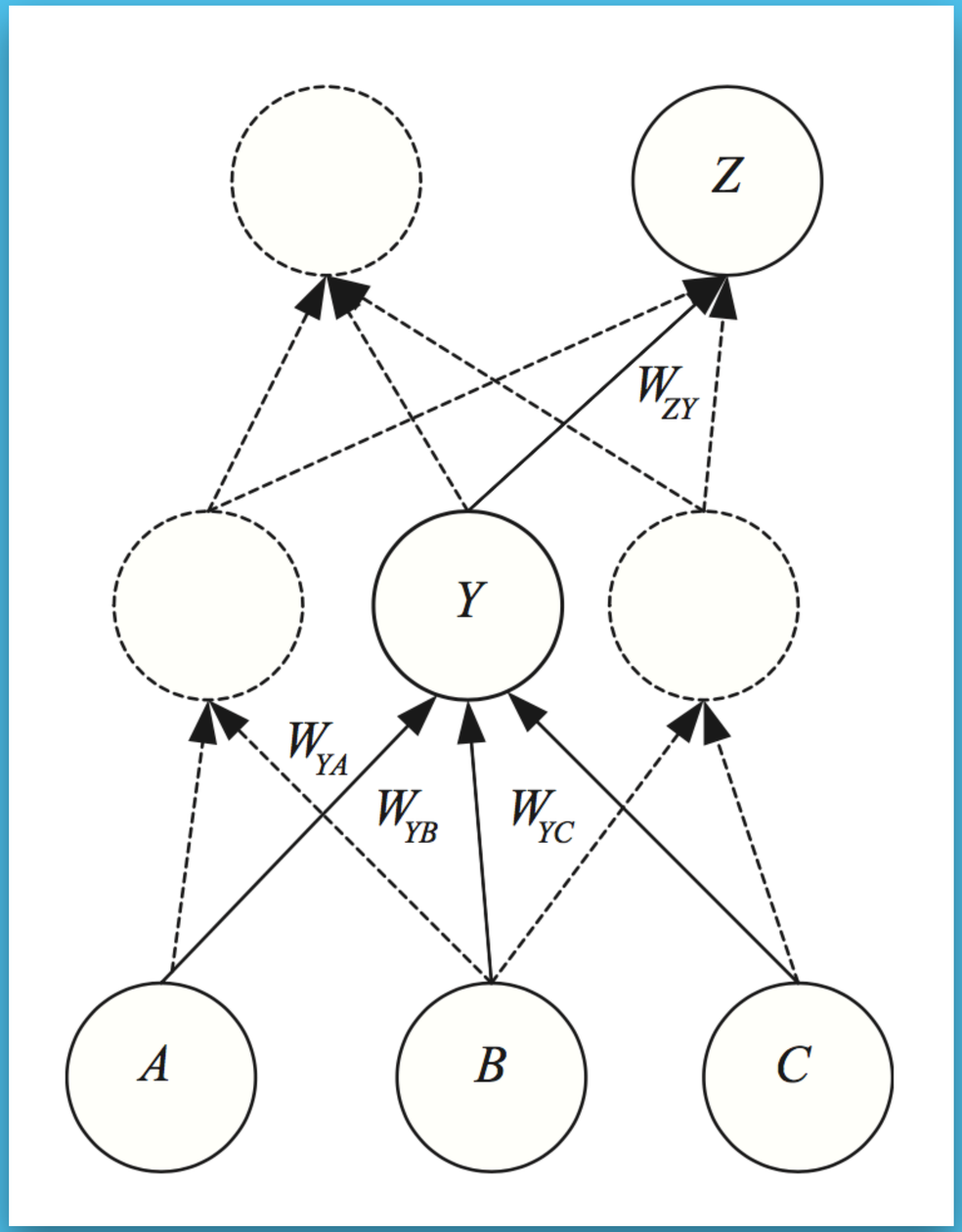
server	The Server on which to run this model.
hidden	An Array that gives the sizes of any hidden layers in the network (default is two hidden layers of three units each).
activation	The activation function to use for the hidden layer units. Beware of the permitted ranges of each: relu (0->inf), sigmoid (0->1), tanh (-1,1).
maxIter	The maximum number of iterations to use in training.
learnRate	The learning rate of the network. Start small, increase slowly.
momentum	The training momentum, default 0.9
batchSize	The training batch size.
validation	The fraction of the DataSet size to hold back during training to validate the network against.

Multilayer Perceptron



- Left: example of a multilayer perceptron (ML), which is a NN with an input layer, output layer, and **may have hidden layers** in between.
- A complex architecture suitable to learn regression and classification models for difficult datasets.

https://deepai.org/machine-learning-glossary-and-terms/multilayer-perceptron



Neural Networks

- Left: example of a neural network (NN) with neurons and synaptic connections.
- Check the glossary file under https://github.com/mirlca/leicesterhackspace-workshop

Alpaydin, Ethem. Machine Learning: The New AI. Cambridge, MA: MIT Press, 2016.

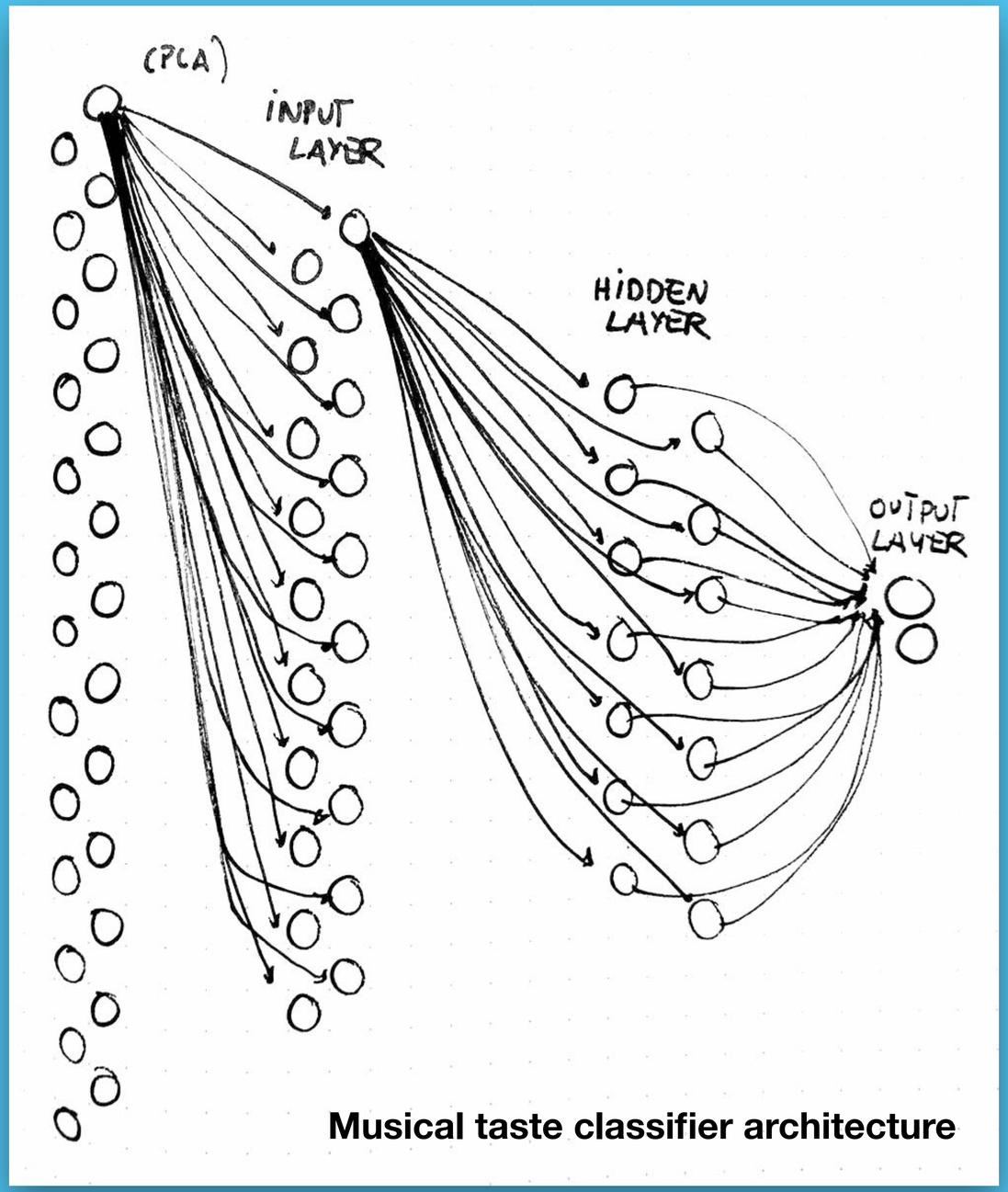
3. Test the implemented ML tasks…

NN-1: Learning my musical taste

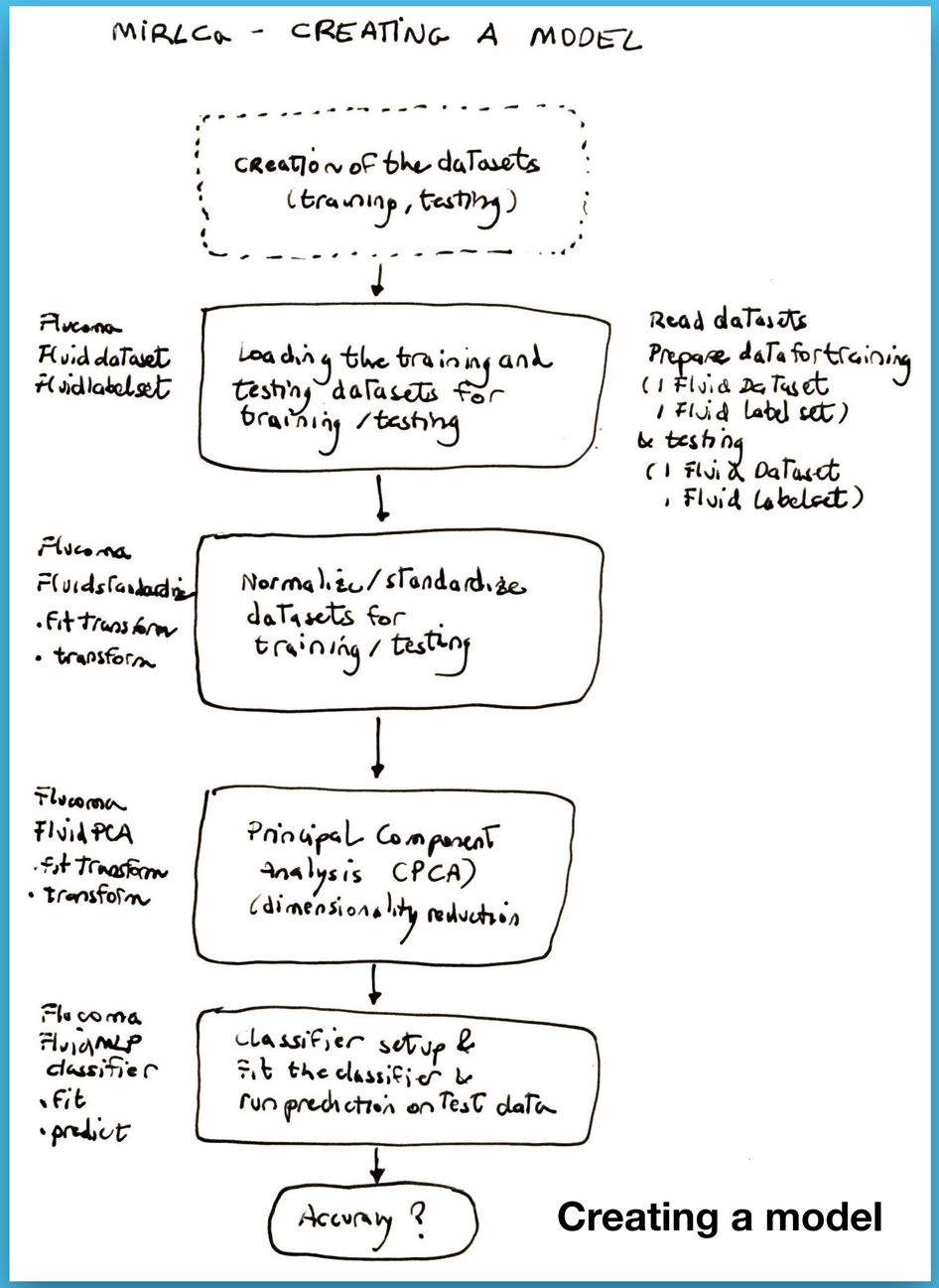
Steps in the training process …

1. For creating the dataset, choose a set of sounds and relevant sound descriptors that characterise the sound samples from Freesound, e.g.:
    - Approach 1: pitch, rhythm (bpm), brightness (spectral centroid), noisiness (spectral flatness)
    - Approach 2: MFCCs
2. Label each sound with “good” or “bad” in an evenly manner.
3. Map these sound descriptors to input layers of the NN and define different parameters for the training.

Detailed blog post at: https://mirlca.dmu.ac.uk/posts/towards-learning-my-musical-taste/



Musical taste classifier architecture



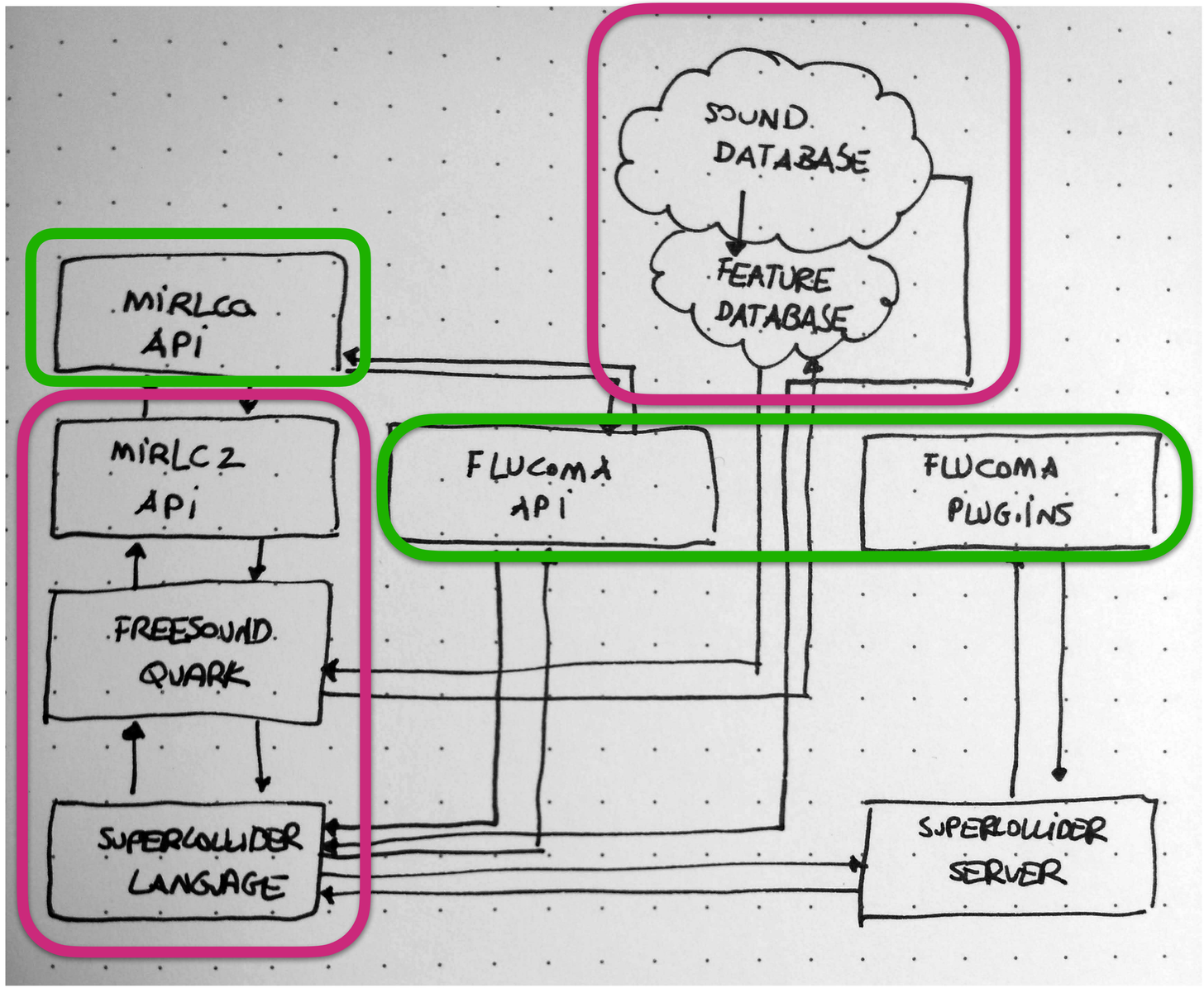
Creating a model

https://github.com/mirlca/code/blob/main/tutorials/manual-training-musical-taste-classifier.scd

Best Descriptors…

- First round with **9 descriptors:**
  - pitch, bpm, centroid, **flatness**, **pitch_confidence** (mean and variance).
  - **73% accuracy with flatness and pitch_confidence (mean and variance)**
- Second round with **26 descriptors:**
  - Mel-frequency cepstral coefficients (MFCCs) (mean and variance) + PCA
  - **76%-83% accuracy**

Next:
Tutorial FluidMLPClassifier
Tutorial MIRLCa Performance
Tutorial MIRLCa Training

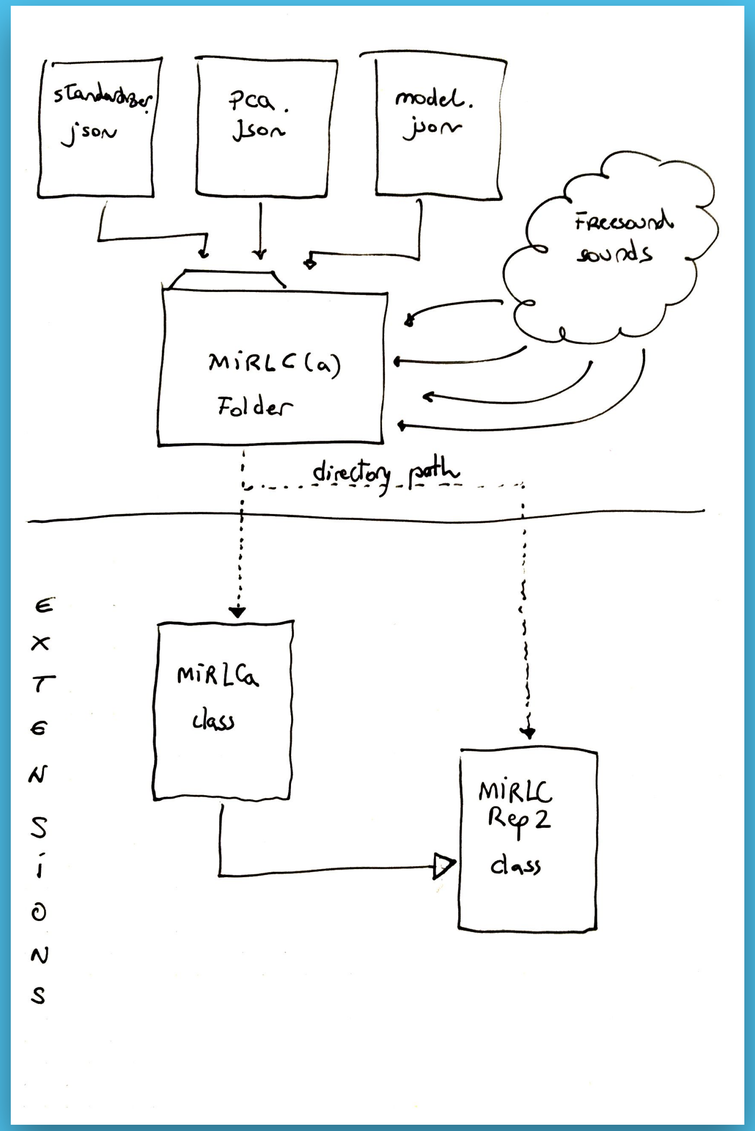


Day 1

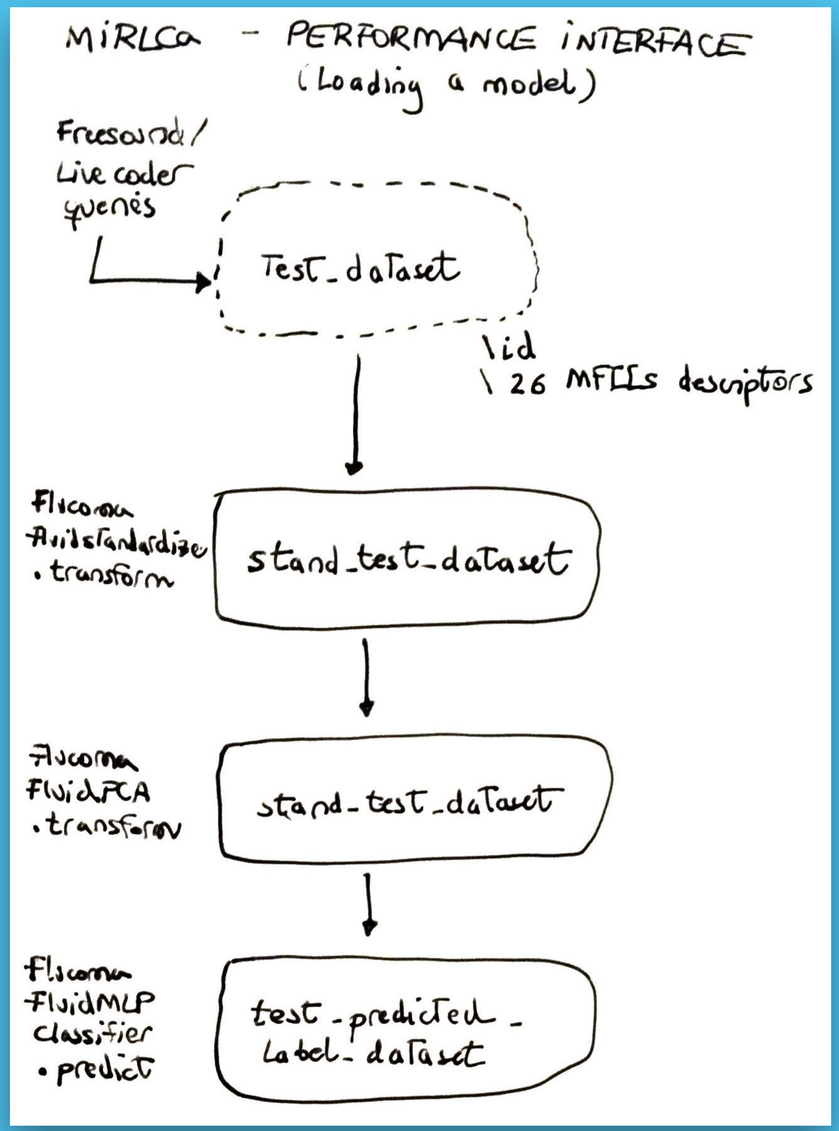
Days 2 & 3

System's architecture

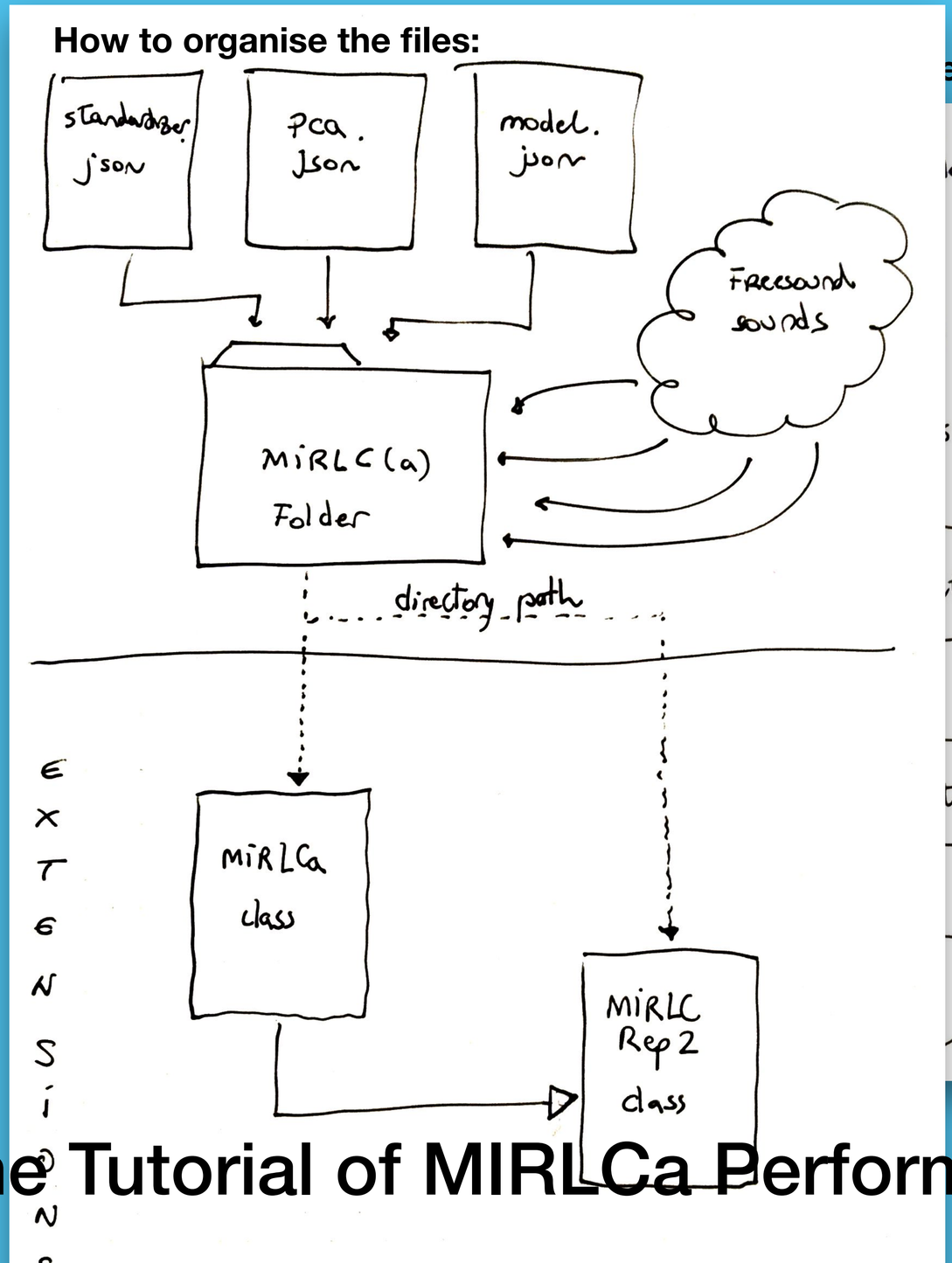
How to organise the files:



Behind the scenes…

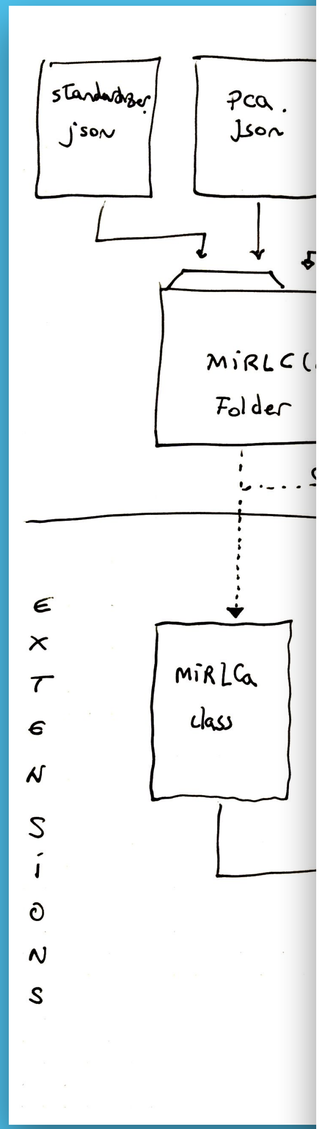


For the Tutorial of MIRLCa Performance

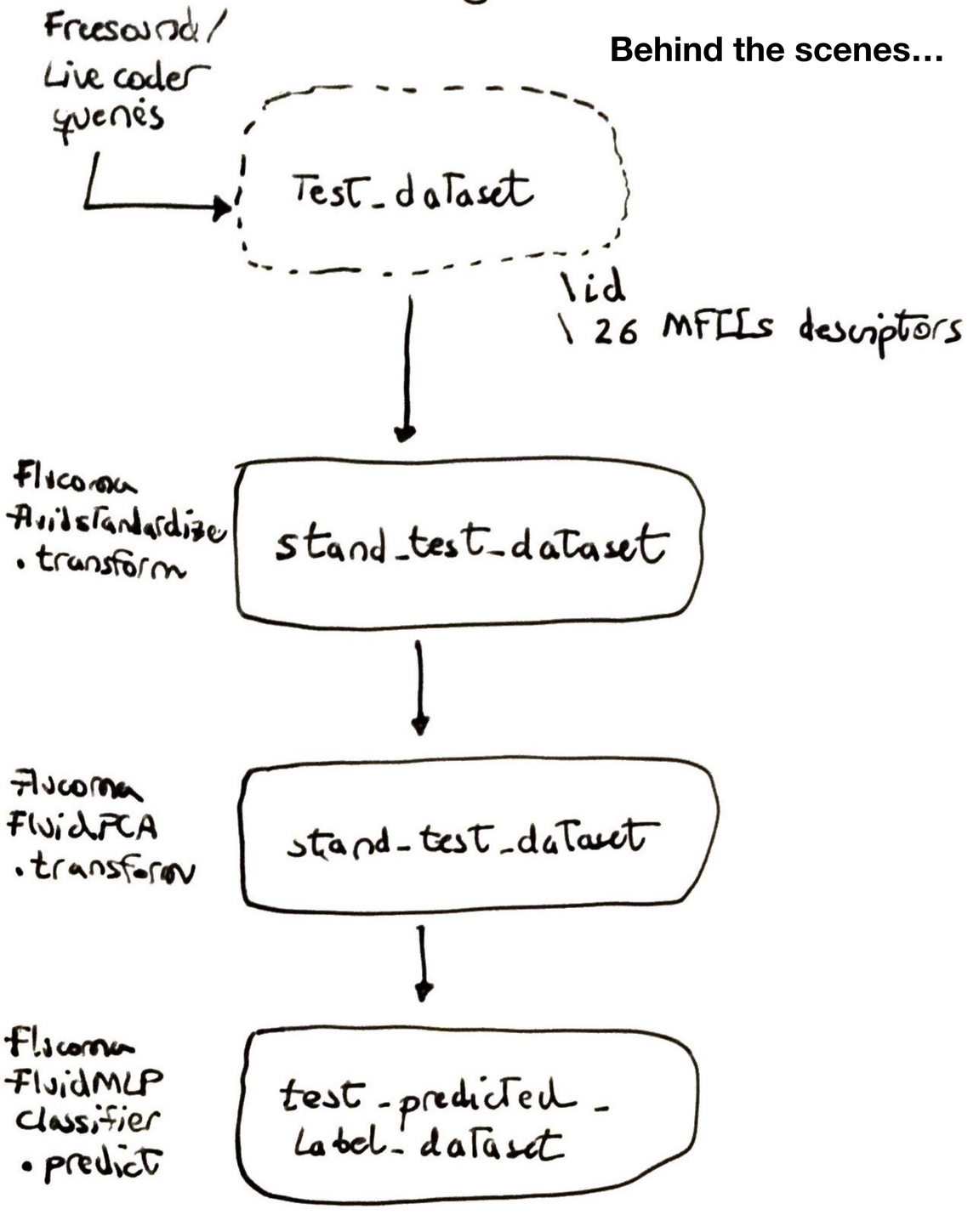


For the Tutorial of MIRLCa Performance

How to or



Behind the scenes…



For th nce

```
// instantiation
~a = MIRLCRep.new
~b = MIRLCRep.new


// GET SOUNDS BY TEXT

// getsound(id=31362, size=1)
~a.id(323399)
~a.id(19246)
~a.id(19247)
~b.id(19248)
~b.id(192468)

// random(size=1)
~a.random
~a.random(2)
~a.random(3)
~b.random

// tag(tag="noise", size=1)
~a.tag("nail", 3)
~a.tag("chimes", 2)
~a.tag("noise", 2)
~a.tag("hammer", 2)
~b.tag("grain", 2)
~b.tag("humming", 3)
```

group ~a

[sound1] [sound2] [sound3] [sound4] [sound5]

index 0 index 1 index 2 index 3 index 4

MIRLCRep

```
// GET SOUNDS BY CONTENT & GET SOUNDS BY CONTENT WITH FILTER

// content(size=1, feature = 'dur', fvalue = 1, fx = 'conf', fxvalue = 'hi')
~a.content // sounds of 1 sec of duration
~a.content(1, 'dur', 10) // sounds of 10 sec of duration
~a.content(1, 'dur', 1, 'key', 'A')
~a.content(1, 'dur', 4, 'scale', 'minor')
~a.content(1, 'dur', 1, 'conf', 'lo')
~a.content(2, 'pitch', 100, 'conf', 'lo')
~a.content(1, 'key', 'Asharp')
~a.content(5, '.lowlevel.spectral_complexity.mean:',1, 'conf', '[0 TO 0.3]') // Using directly Essentia's format
~b.content(1, 'bpm', 120)
```

```
// GET SIMILAR SOUNDS BY EXAMPLE

// similar(targetnumsnd=0, size=1)

~a.similar
~a.similar(0)
~a.similar(0, 2)
~b.similar(1)
```

group ~a

[sound1] [sound2] [sound3] [sound4] [sound5]

index 0 index 1 index 2 index 3 index 4

```
// GET SIMILAR SOUNDS BY FILTER

// filter (targetnumsnd=0, size=1, fx = 'conf', fxvalue = 'hi')

~a.content(1, 'dur', 4, 'scale', 'minor')
~a.filter(1, 1,'conf','lo')
~a.filter(1, 1,'conf','hi')
~a.filter(2, 1,'conf','hi')

~b.content(1, 'dur', 2)
```

MIRLCRep 1.0

References

- Great blog on ML: https://machinelearningmastery.com

- Alpaydin, Ethem. *Machine Learning: The New A*I. Cambridge, MA: MIT Press, 2016.