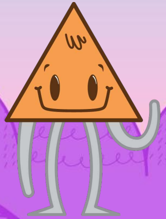
ml5.js: Friendly Machine Learning for the Web

Yining Shi
Adjunct professor at New York University
Software Engineer at RunwayML



Friendly Machine Learning for the Web

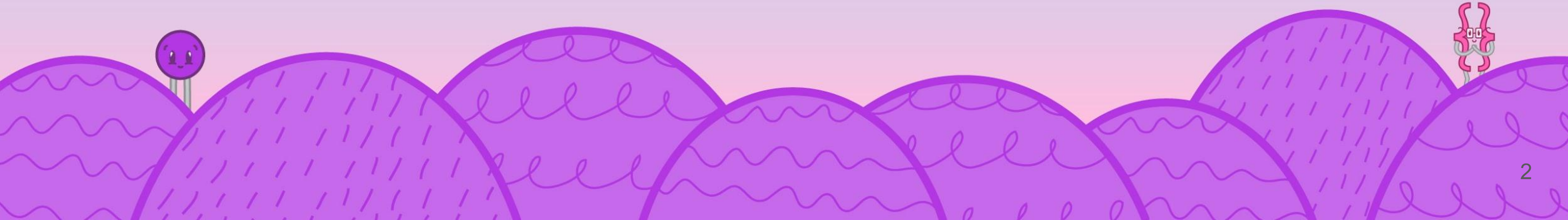
A neighborly approach to creating and exploring artificial intelligence in the browser.

Developing ml5 is not just about developing machine learning software, it is about making machine learning approachable for a broad audience of **artists, creative coders, and students**.

The library provides access to machine learning algorithms and models in the browser, building on top of **TensorFlow.js**.

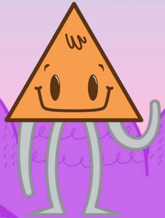
ml5.js is heavily inspired by Processing and p5.js.

ml5js.org



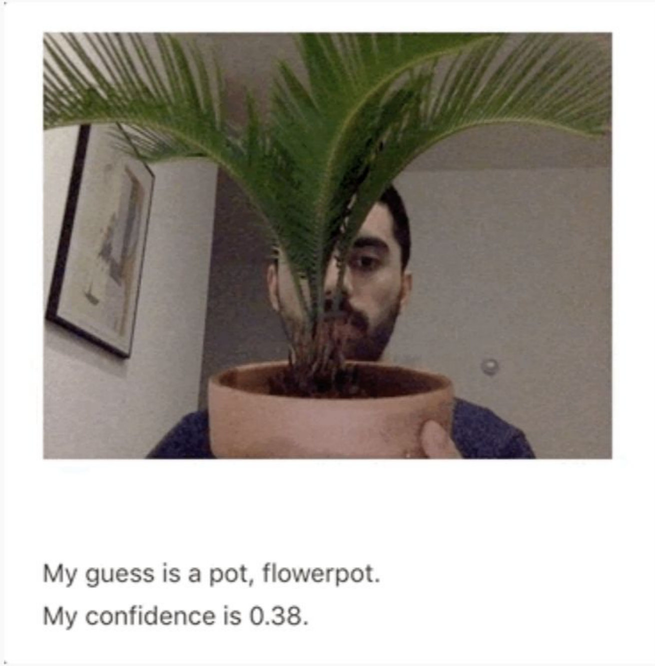
What does ml5.js do?

1. Running pre-trained models
2. Creating and training neural networks



imageClassifier(‘MobileNet’)

ported by Cristobal Valenzuela



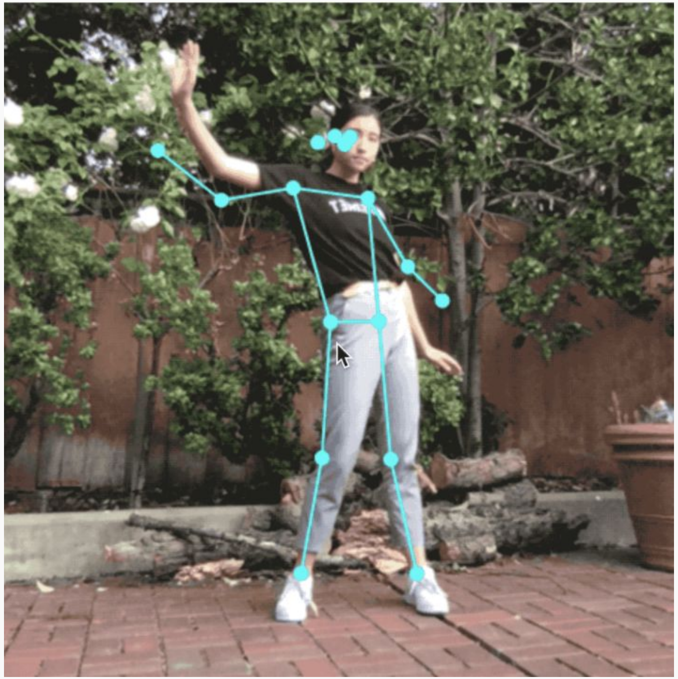
My guess is a pot, flowerpot.

My confidence is 0.38.

```
const classifier = ml5.imageClassifier(‘MobileNet’);

classifier.classify(video, gotResult);

function gotResult(error, result) {
  console.log(result);
}
```

PoseNet

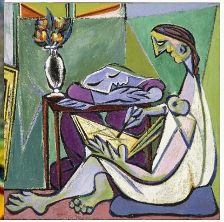
ported by Cristobal Valenzuela, Maya Man, Dan Oved.



```
const posenet = ml5.poseNet(video);

posenet.on('pose', function(results) {
  poses = results;
});

function draw() {
  if (poses.length > 0) {
    circle(poses[0].nose.x, poses[0].nose.y);
  }
}
```

Style Transfer

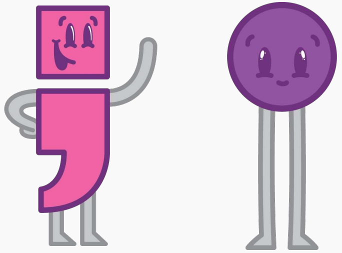
ported by Yining Shi

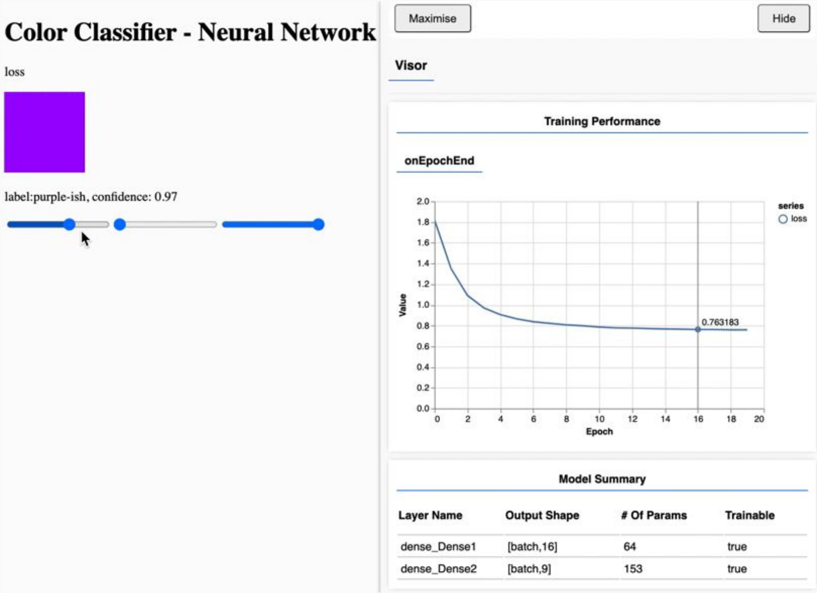


```
const cubist = ml5.styleTransfer(‘models/cubist’, modelReady);

function modelReady() {
  cubist.transfer(video, gotImage)
}

function gotImage(error, result) {
  image(result.image, 0, 0);
}
```

DIY Neural Network

Wrapped by Dan Shiffman, Joey Lee & Yining Shi





Color Classifier - Neural Network

loss

label:purple-ish, confidence: 0.97

Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,16]	64	true
dense_Dense2	[batch,9]	153	true

```
const options = {
  dataUrl: 'data/colorData.json',
  inputs: ['r', 'g', 'b'],
  outputs: ['label'],
  task: 'classification'
};

// Create Neural Network
neuralNetwork = ml5.neuralNetwork(options);

neuralNetwork.train(whileTraining, finishedTraining);

neuralNetwork.classify(data, gotResults);
```

image

- imageClassifier
- ObjectDetector
- poseNet
- BodyPix
- UNET
- YOLO
- StyleTransfer
- Pix2Pix
- CartoonGAN
- FaceApi
- Facemesh
- Handpose
- CVAE
- DCGAN
- SketchRNN

sound

- Pitch Detection
- SoundClassifier

text

- CharRNN
- Word Vectorization
- Sentiment
- UniversalSentence
- Encoder

helpers

- NeuralNetwork
- FeatureExtractor
- KNNClassifier
- KMeans

ml5.js version 0.5.0 https://ml5js.org

How do we use ml5.js?

Step 1: Call your ml5 function.

```
const myClassifier = await ml5.imageClassifier(‘MobileNet’);
```

Step 2: Apply your ml5 function - e.g. to an image, video, or text.

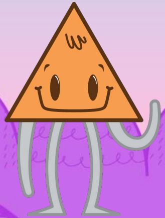
```
const results = await myClassifier.classify(myCatImage);
```

Step 3: Do something with the results.

```
// An array of objects with “label” and “confidence”
// [ { label: ‘cat’, confidence: 0.74 } ]
console.log(results);
```

ml5.js in 3 simple steps

How is ml5.js built?



Core

ml5-library

ml5-examples

Education / Training / models

training-*

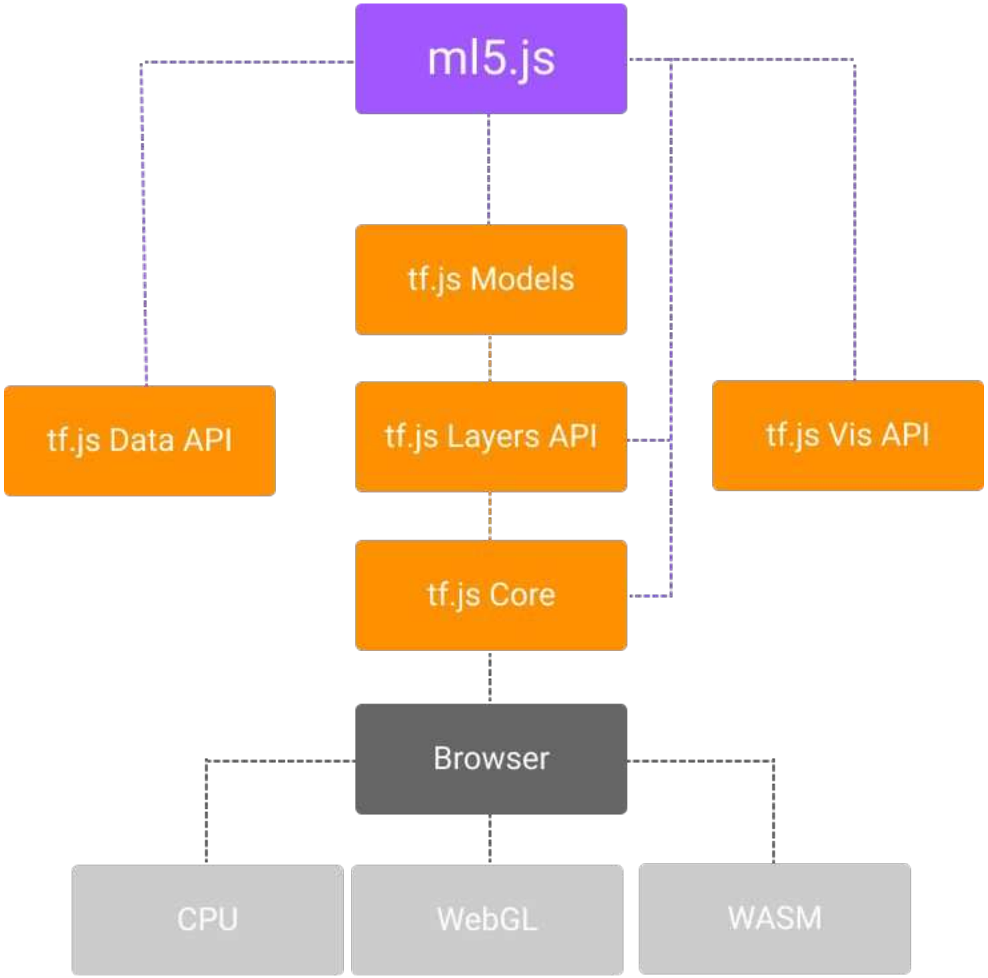
ml5-website

intro-to-ml-arts

ml5-data-and-models-server

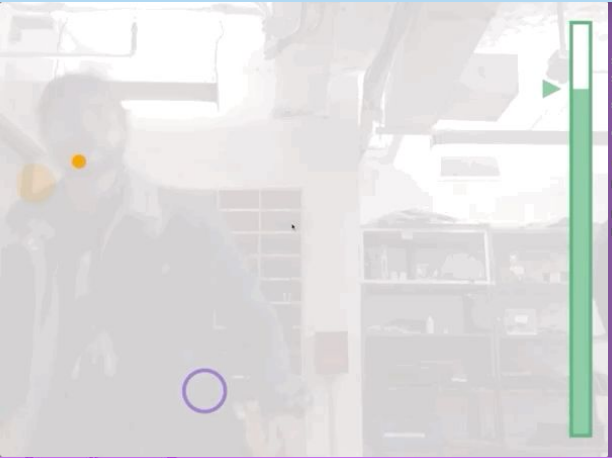
ml5 ecosystem

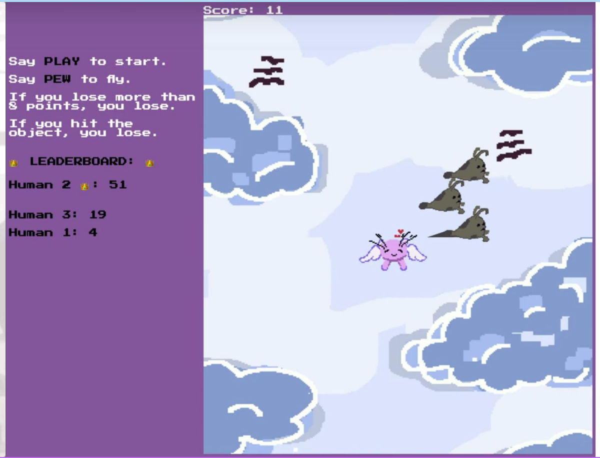
https://github.com/ml5js


ml5.js
tf.js Models
tf.js Data API
tf.js Layers API
tf.js Vis API
tf.js Core
Browser
CPU
WebGL
WASM

The case for ML in the browser:

- Accessibility + familiarity of web apps
- Ease of creation of interactive use cases
  - Low effort in using existing browser APIs such as webcams and microphones with the ability of rendering outputs easily to image, canvas, audio, or text elements on the DOM.





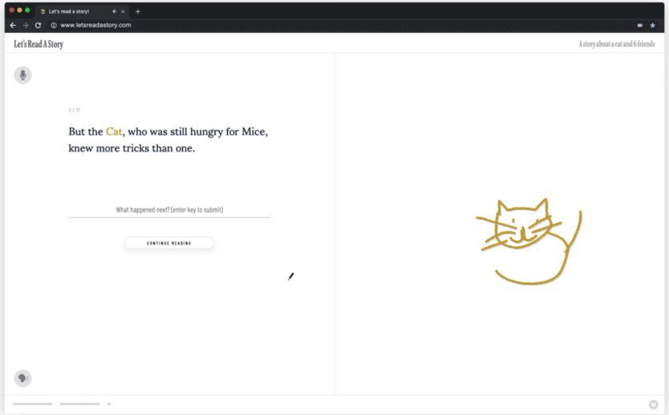
Score: 11

Say PLAY to start.
Say PEW to fly.
If you lose more than 8 points, you lose.
If you hit the object, you lose.

LEADERBOARD:

Human 2 : 51

Human 3: 19
Human 1: 4



Let's Read A Story

A story about a cat and 6 friends

3 / 17

But the Cat, who was still hungry for Mice, knew more tricks than one.

What happened next? (enter key to submit)

https://ml5js.org/community

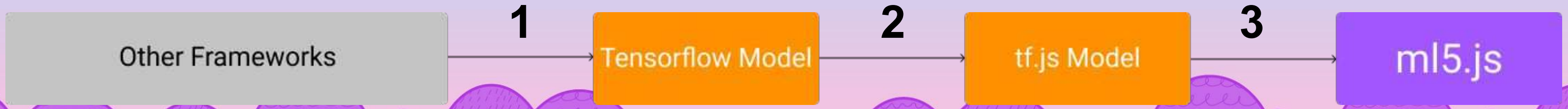
Challenge 1: Native support for converting browser I/O streams to model input/output data structures

- Accessibility + familiarity of web apps
- Ease of creation of interactive use cases
  - Better support for real-time interaction with webcam, microphones as inputs and image, canvas, audio, text as outputs

https://ml5js.org/community

Challenge 2: Standardization of model format

- Factors to determine whether pre-trained models can be ported to ml5.js:
  - Small model size
  - Low-latency
  - Portable model format
- Workflow of porting:



Other Frameworks → 1 → Tensorflow Model → 2 → tf.js Model → 3 → ml5.js

Thank you!

https://ml5js.org
https://github.com/ml5js
Twitter @ml5js