

ARTIFICE - AI workshop

Data, Algorithm & Computation, Creativity

Janet Huang

2023.10.11

Objectives of this AI workshop

- Understand how to turn your idea into practice
 - Understand how to use tools for supporting consolidating your idea
 - Understand how to use tools/libraries/algorithms for supporting prototyping with data and AI
- Build an AI (i.e., an image/audio/pose classifier) based on **your own data**
- Build an AI (i.e., an object detector) with a **pre-trained model** to recognize 80 everyday objects
- Design/build an entire system, as a **blueprint**, for realizing your project idea

Workshop Structure

- **Session 0:** Reflect on your mini assignment **(15 mins)**
 - build an image classifier using Teachable Machine
- **Session I:** Design your AI agents **(25 mins)**
 - MS-COCO dataset
 - thingCV tool
 - exercise 1: design your agent with AI canvas

- **Session II:** Build your own thingCV
 - introduction to p5.js, ml5.js and Data Foundry **(10 mins)**
 - exercise 2: object detector **(data foundry, p5.js, ml5.js) (30 mins)**

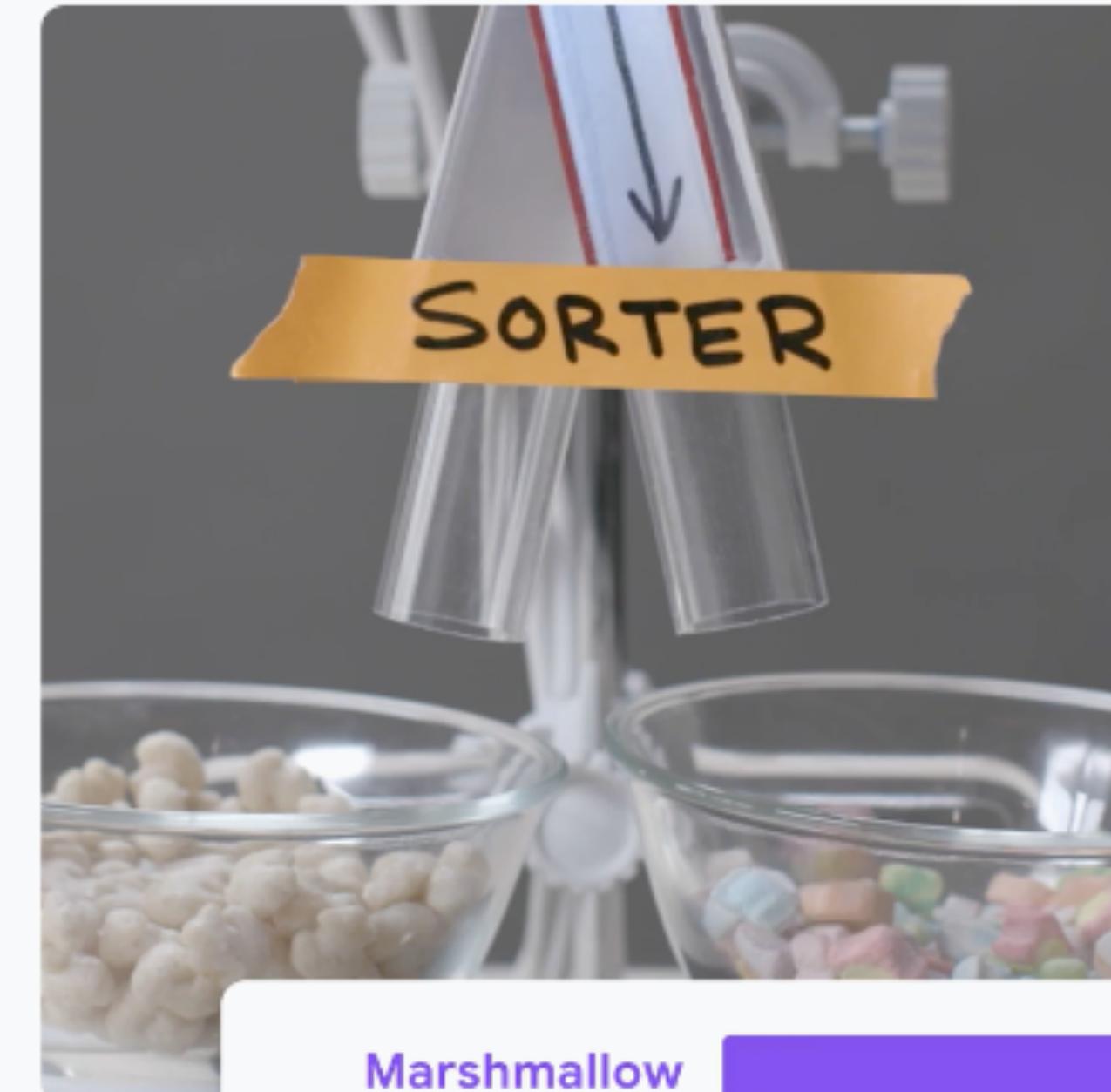
- exercise 3: store your data in DF **(data foundry, IoT dataset) (20 mins)**
- exercise 4: data visualization **(d3.js, data foundry) (20 mins)**
- **Recap**

Session 0: Teachable Machine

Teachable Machine

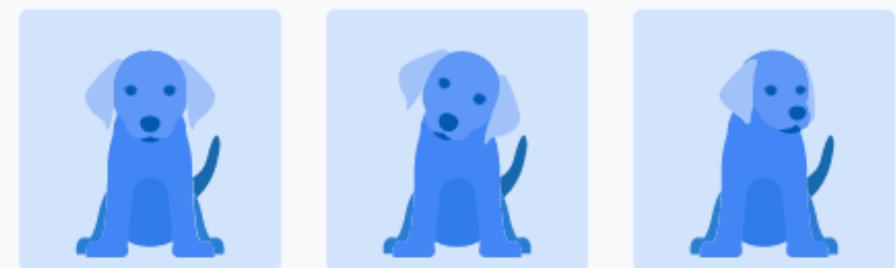
Train a computer to recognize your own images, sounds, & poses.

A fast, easy way to create machine learning models for your sites, apps, and more – no expertise or coding required.

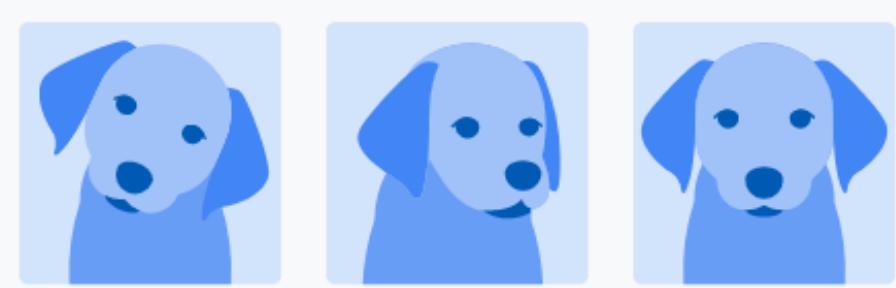
[Get Started](#)

Teachable Machine

Class 1



Class 2



1 Gather

Gather and group your examples into classes, or categories, that you want the computer to learn.

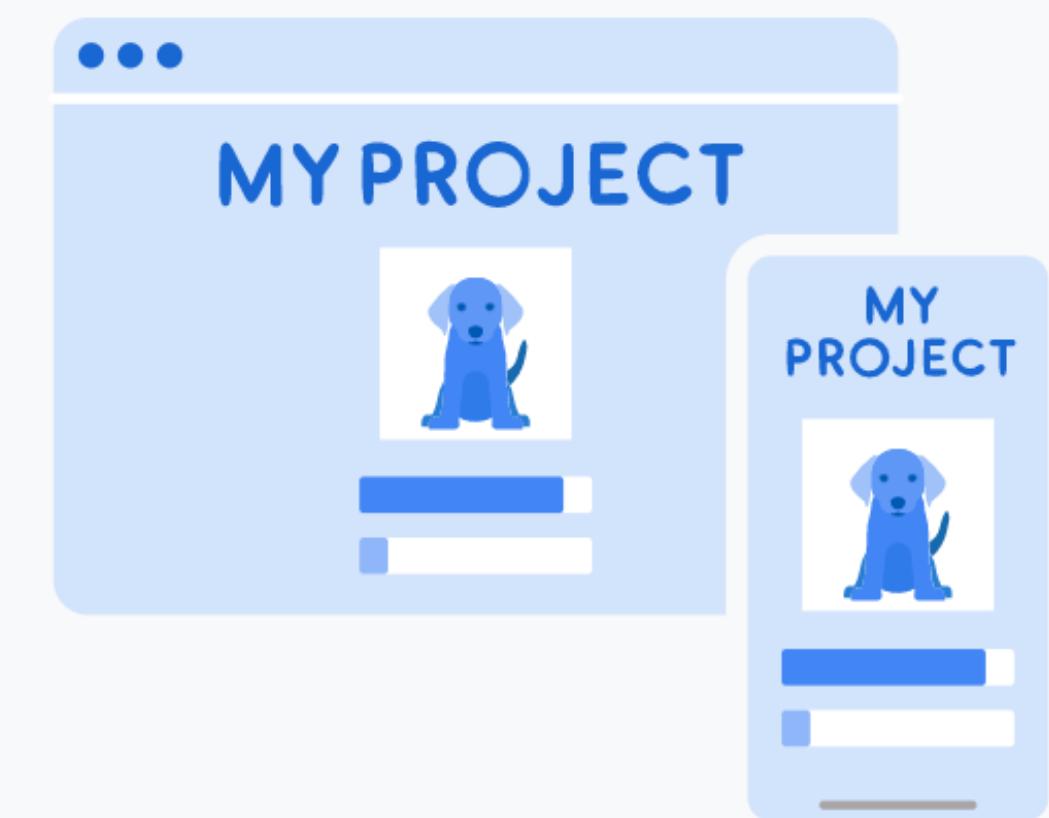
2 Train

Train your model, then instantly test it out to see whether it can correctly classify new examples.



3 Export

Export your model for your projects: sites, apps, and more. You can download your model or host it online for free.



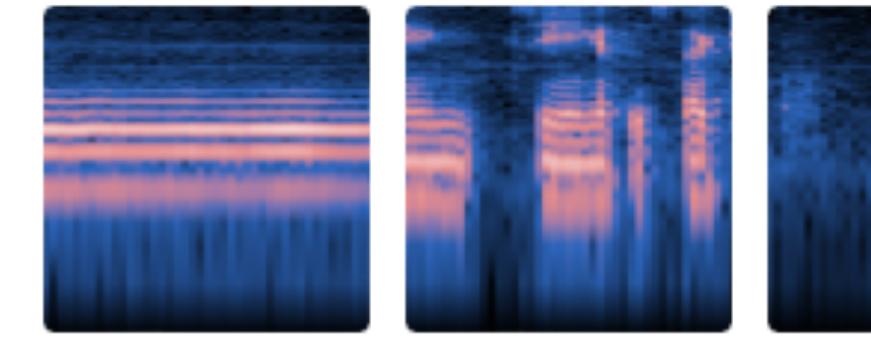
Warm-up Exercise: (5 mins)

Play with the model trained from your peers



Image Project

Teach based on images, from files or your webcam.



Audio Project

Teach based on one-second-long sounds, from files or your microphone.



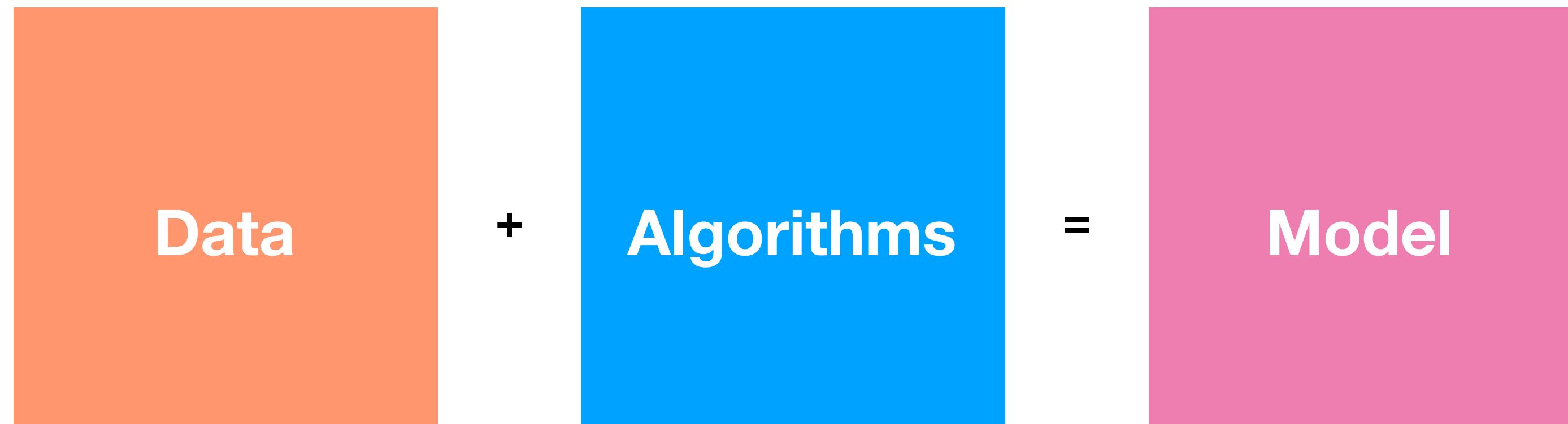
Pose Project

Teach based on images, from files or your webcam.

Share your reflection with us!

AI: What is inside?

- Data
 - raw data vs labels (i.e., human annotation)
- Feature
- Parameters
- Algorithms
- Model



Opportunities to consider
“perception differences”
encoded in data

a lot of focus on explaining
“algorithmic reasoning”

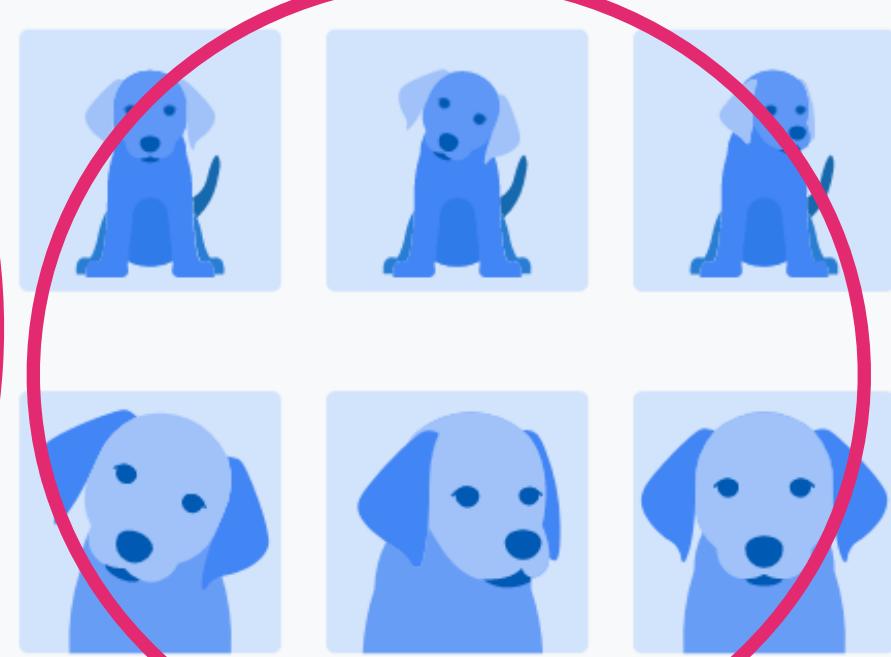
Machine Intelligence

Teachable Machine

(class) labels

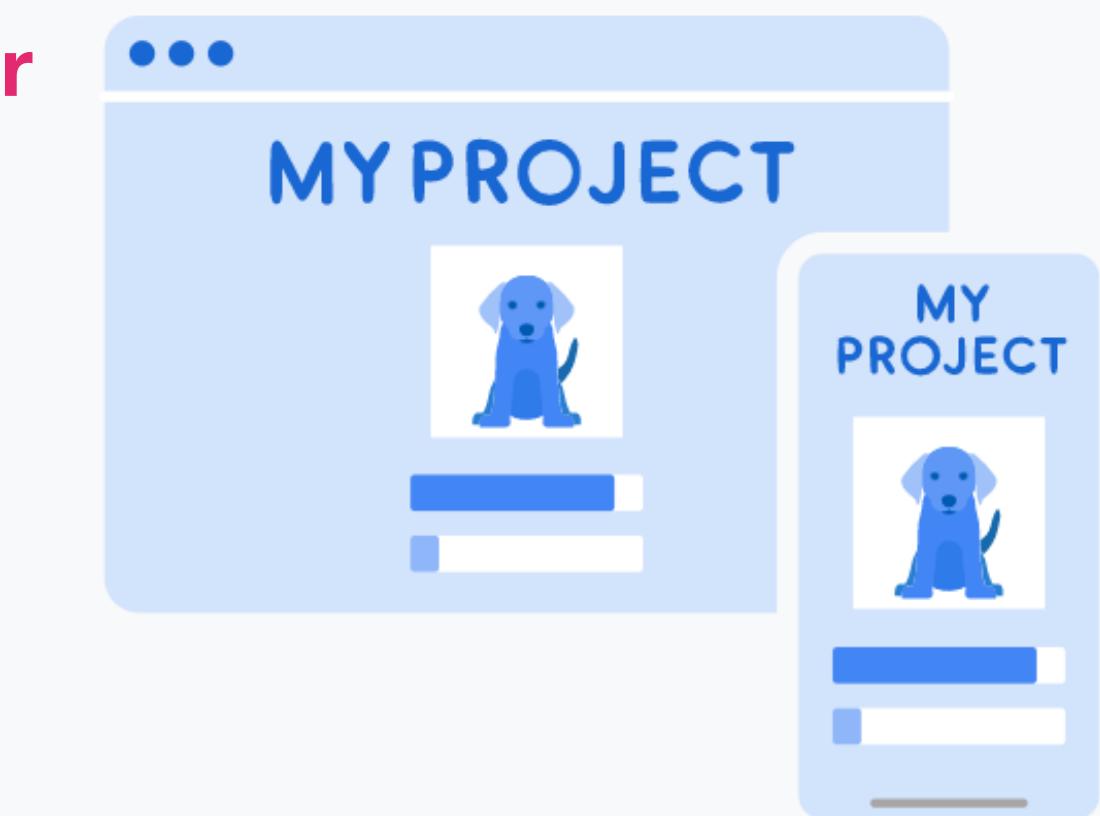
Class 1
Class 2

Raw data (unlabeled data)



(image, audio, or pose) classifier

TRAIN MODEL



1 Gather

Gather and group your examples into classes, or categories, that you want the computer to learn.

2 Train

Train your model, then instantly test it out to see whether it can correctly classify new examples.

3 Export

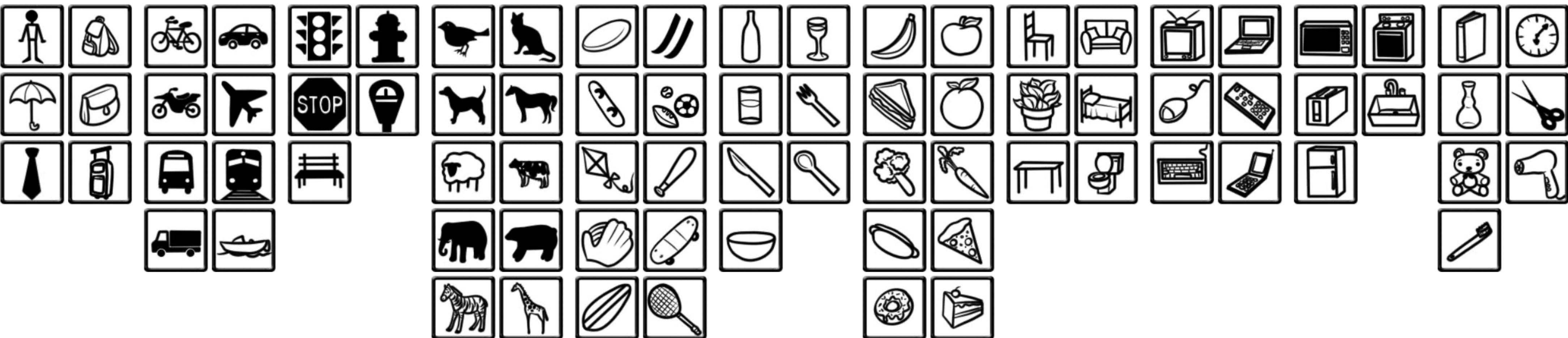
Export your model for your projects: sites, apps, and more. You can download your model or host it online for free.

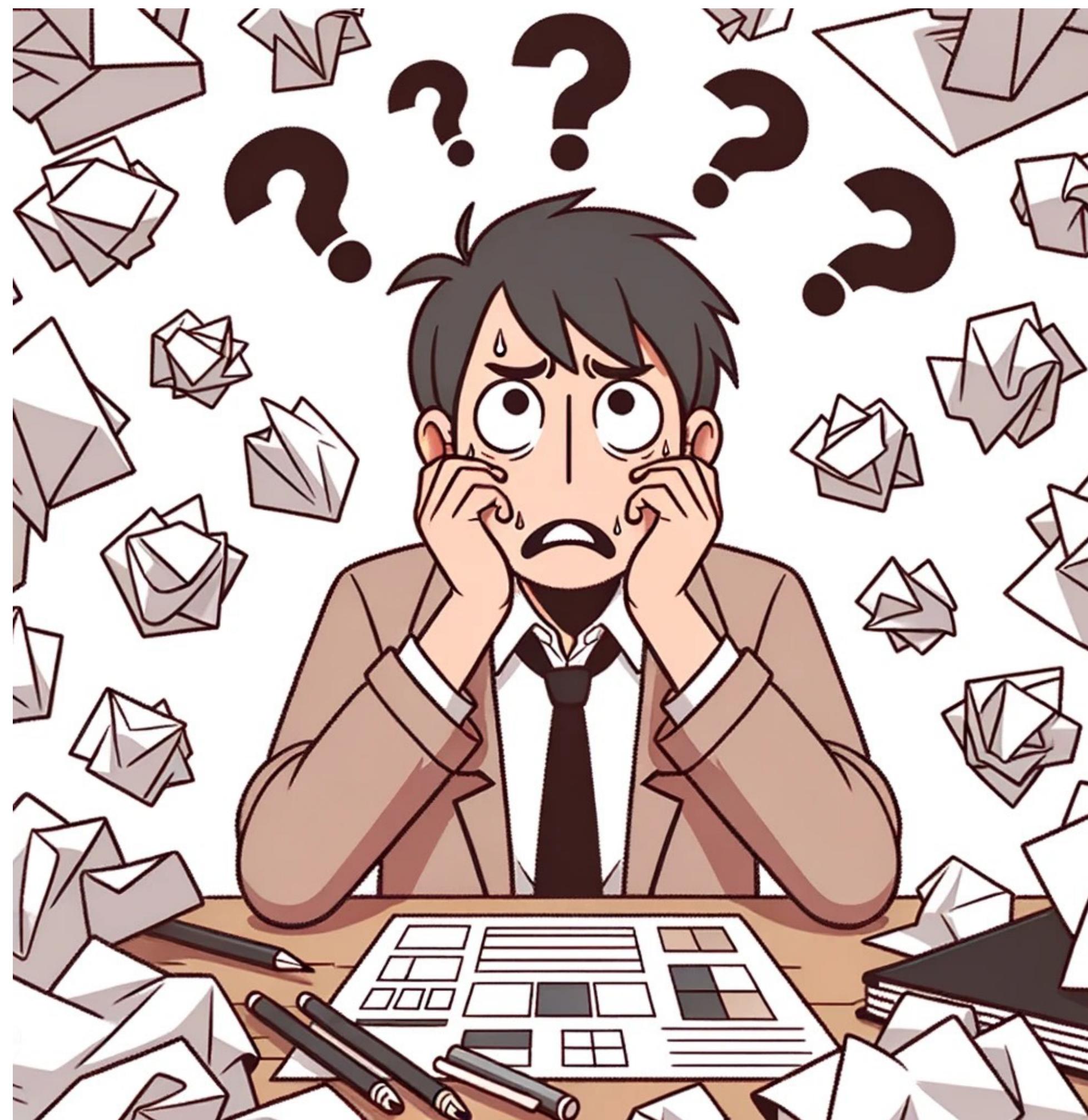
algorithm: “transfer learning”

Session I: Design your AI agents

Task: Design Your AI Agents

Pickup one object and redesign it as an AI agent in a IoT ecosystem



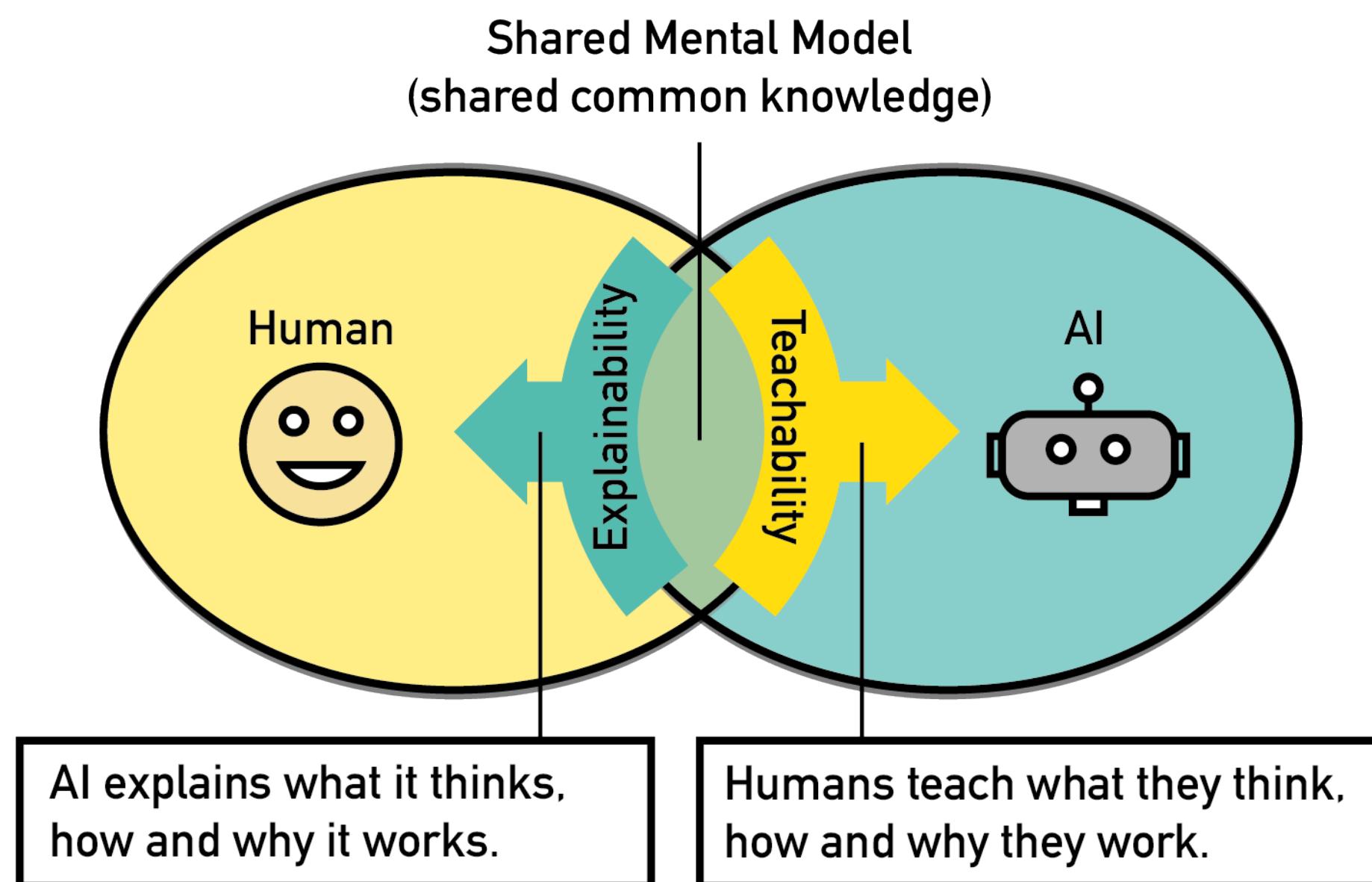


Prompt: Illustration of a designer with a worry face, surrounded by crumpled design sketches, symbolizing his confusion and uncertainty about his work. Created by DALL-E 3

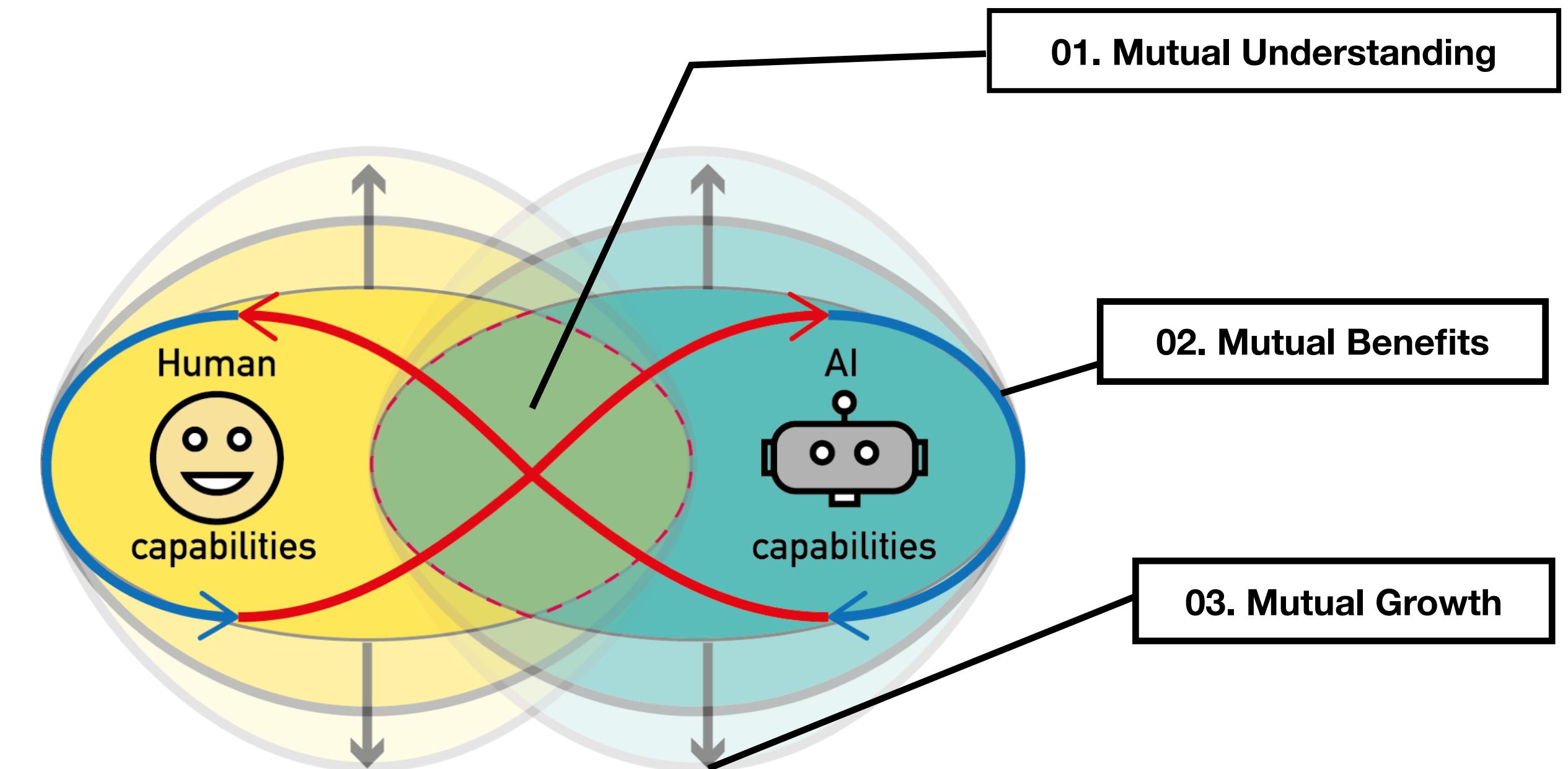


Prompt: Illustration of a female designer with a concerned expression, holding her head in her hands, amidst a chaotic workspace filled with design drafts and tools. Created by DALL-E 3

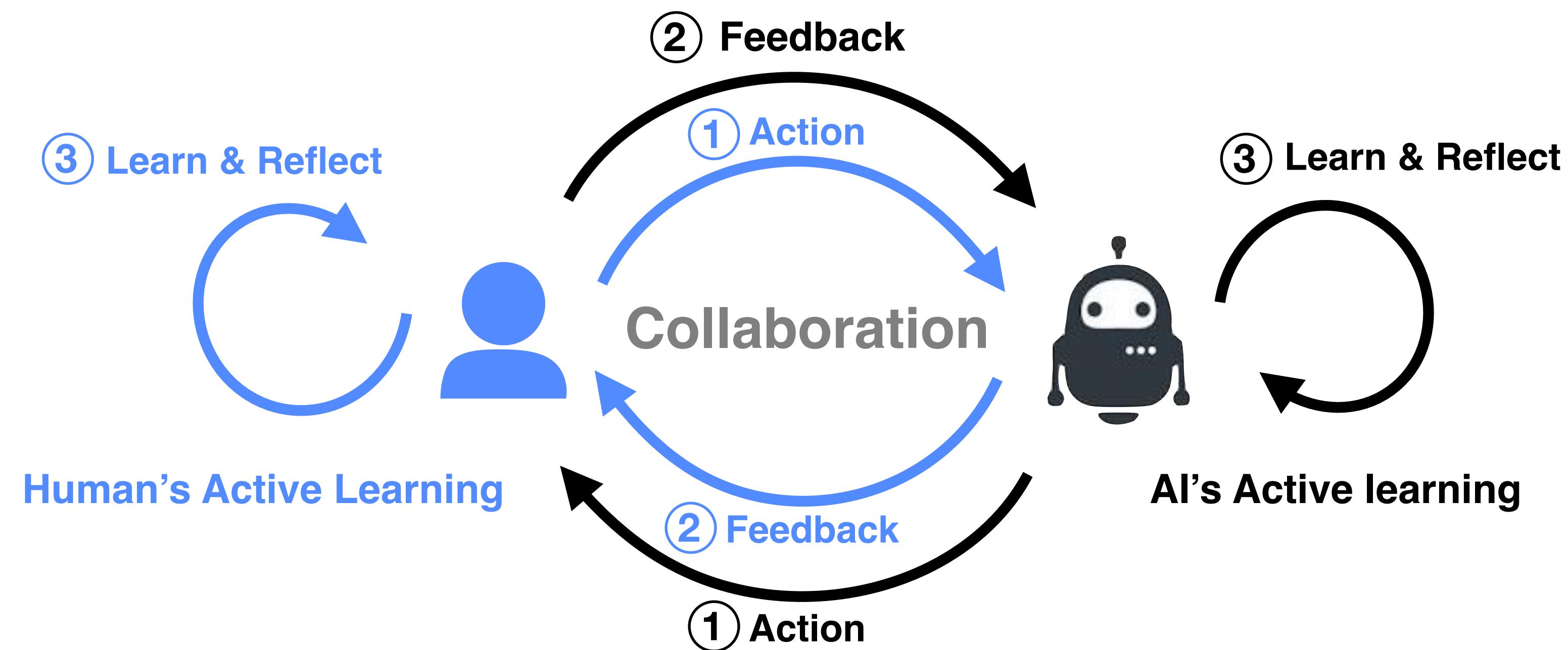
Human-AI Interaction



Human-AI Co-Learning

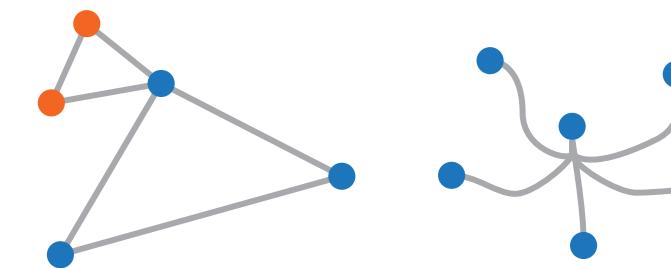


Human-AI Collaboration Framework

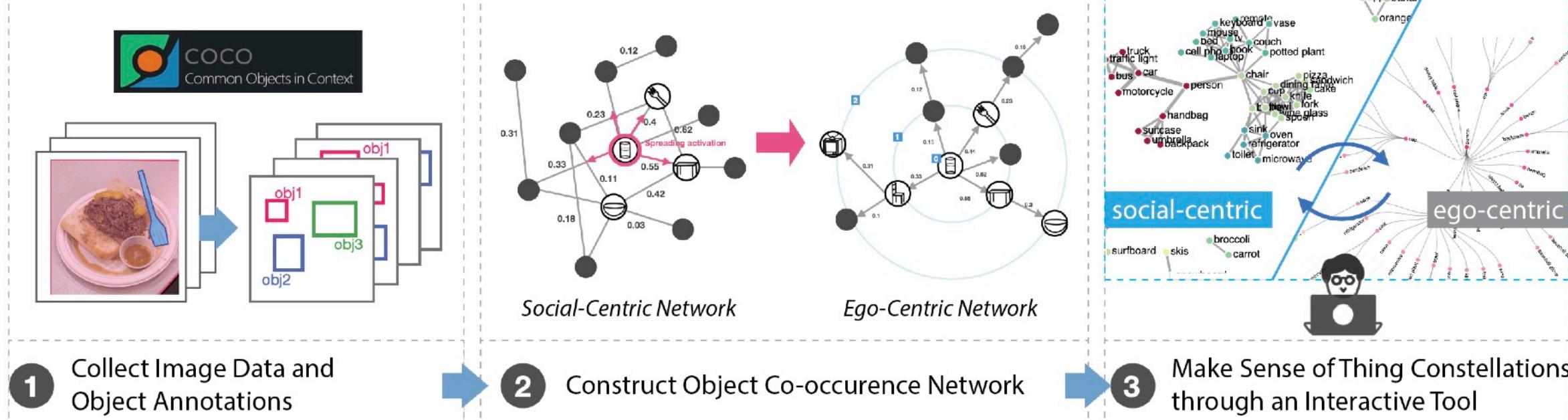


Thing Constellation Visualizer

Exploring Emergent Relationships of Everyday Objects



Data-Driven Design Exploration



Data-driven pattern + Human interpretation

Contributions

This work presents Thing Constellation Visualizer (ThingCV), a new approach and tool that empowers designers to use alternative perspectives to revisit everyday practice and contributes new insights from two workshops on in-depth understandings of emergent relationships among objects.

The approach, tool, and insights will contribute to the future design of IoT ecosystems.



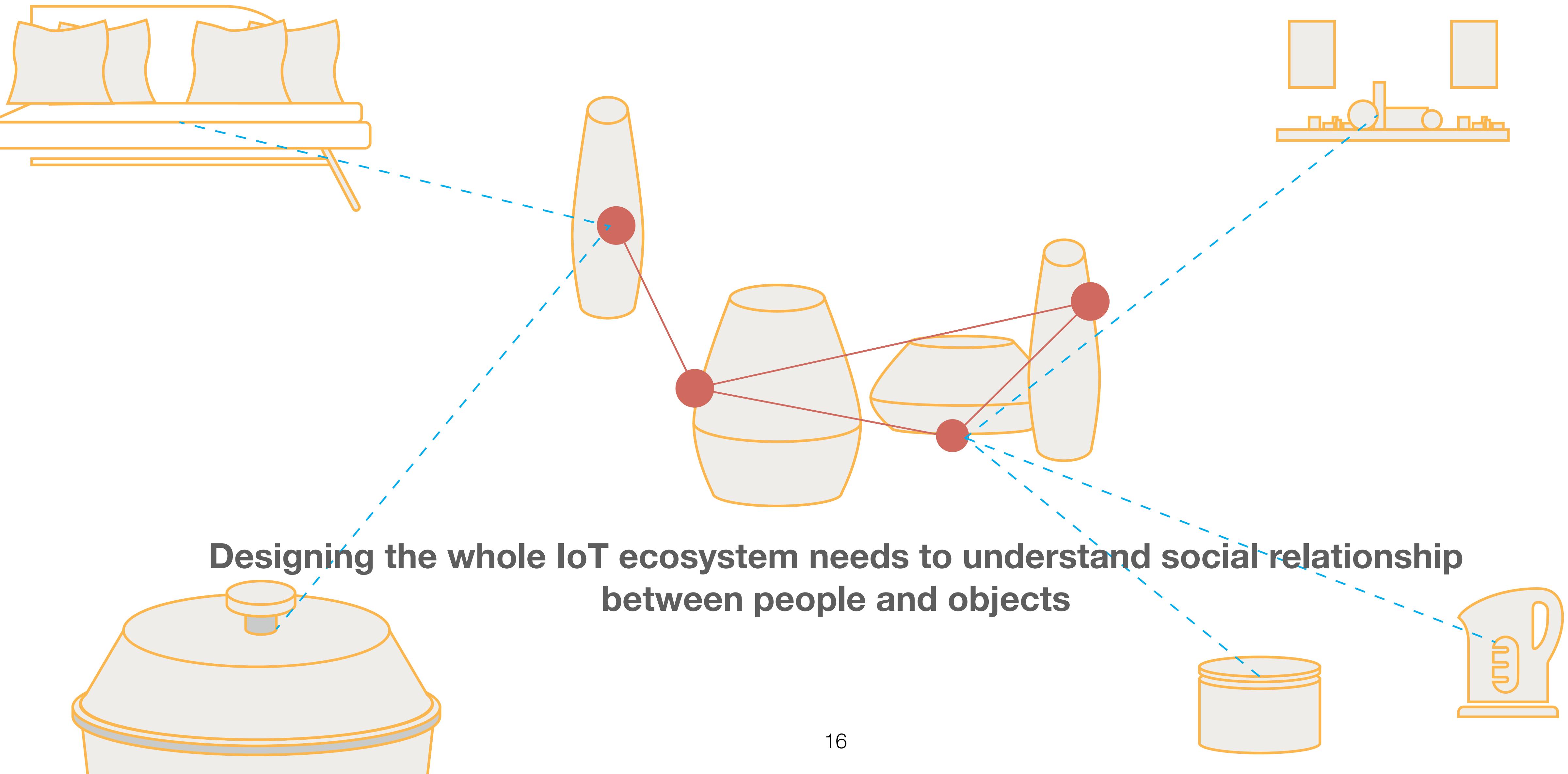
Janet Huang
TU Eindhoven

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

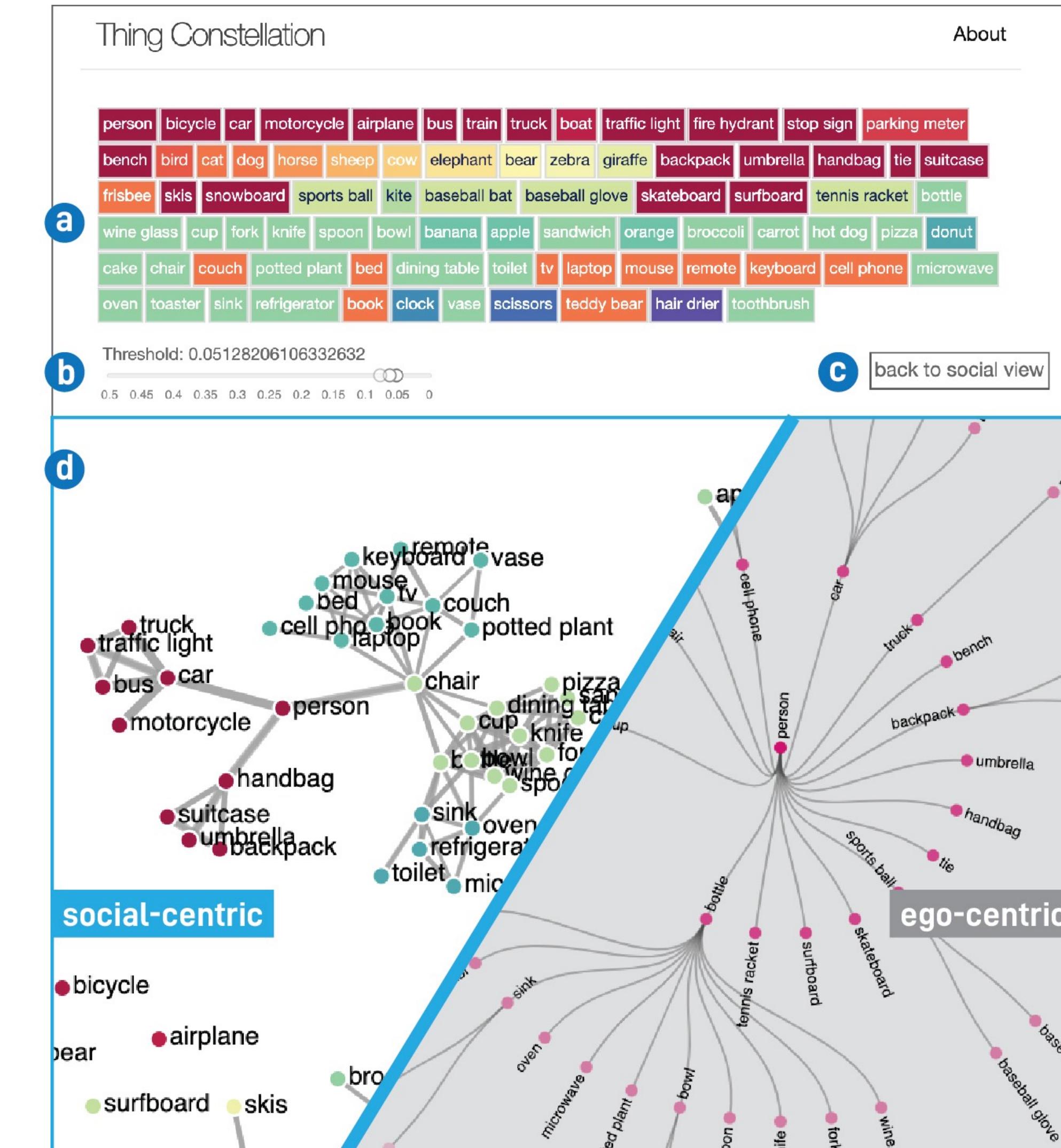
Yi-Ching (Janet) Huang, Yu-Ting Cheng, Rung-Huei Liang, Jane Yung-jen Hsu, and Lin-Lin Chen. 2021. Thing Constellation Visualizer: Exploring Emergent Relationships of Everyday Objects. Proc. ACM Hum.-Comput. Interact. 5, CSCW2, Article 479 (October 2021), 29 pages. DOI:<https://doi.org/10.1145/3479866>

thingCV (<https://thingconstellation.github.io>)





Thing Constellation Visualizer



a Object Panel

- objects in the same community are filled with the same colors.
- click any object to zoom into ego-centric view.

b Threshold Slider

- adjust threshold value to determine the density (i.e. numbers of links between nodes).

c Switch Button

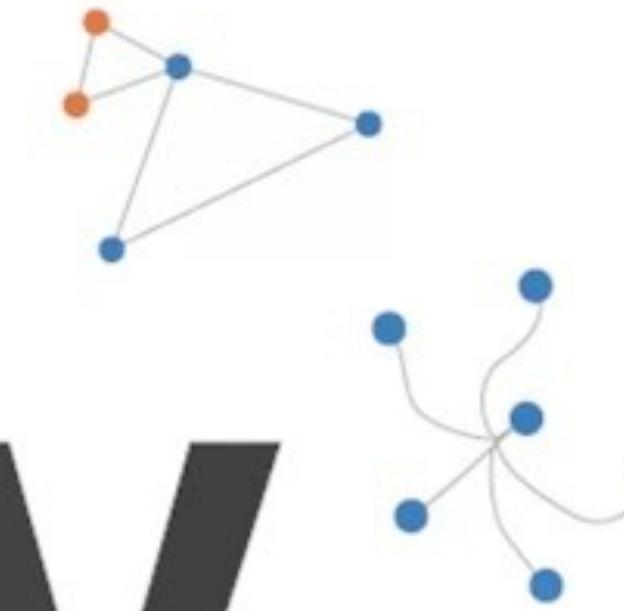
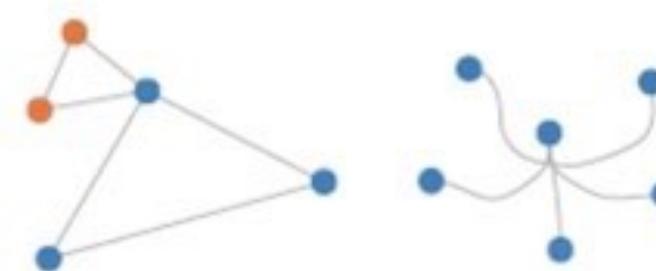
- switch two views under the same threshold value.

d Thing Constellation Visualisation

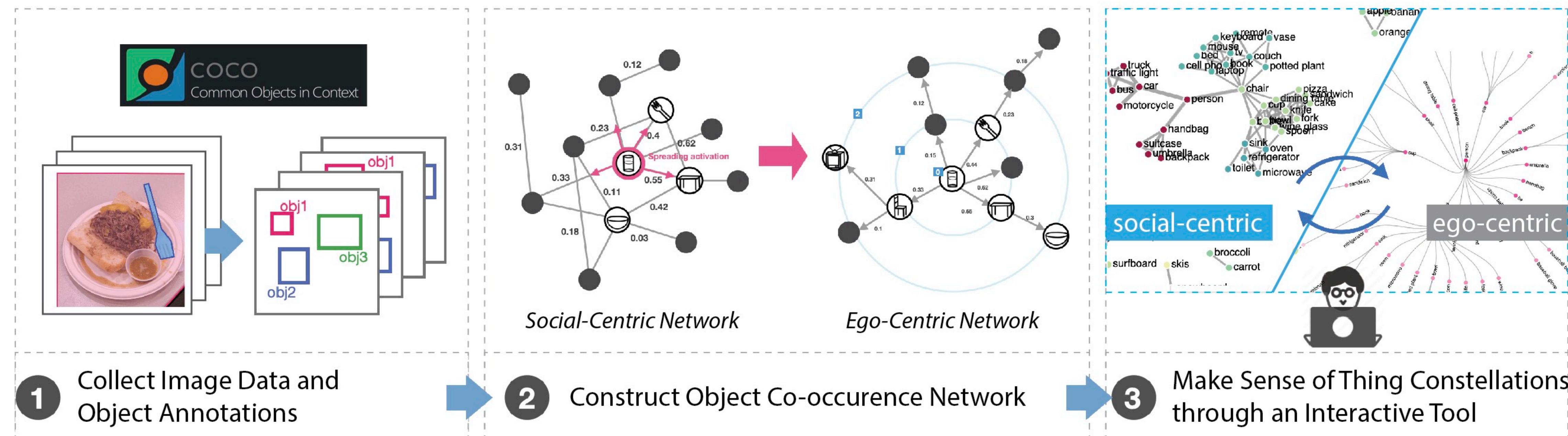
- based on the previous selections, here shows the result which can be social-centric or ego-centric constellation.

ThingCV

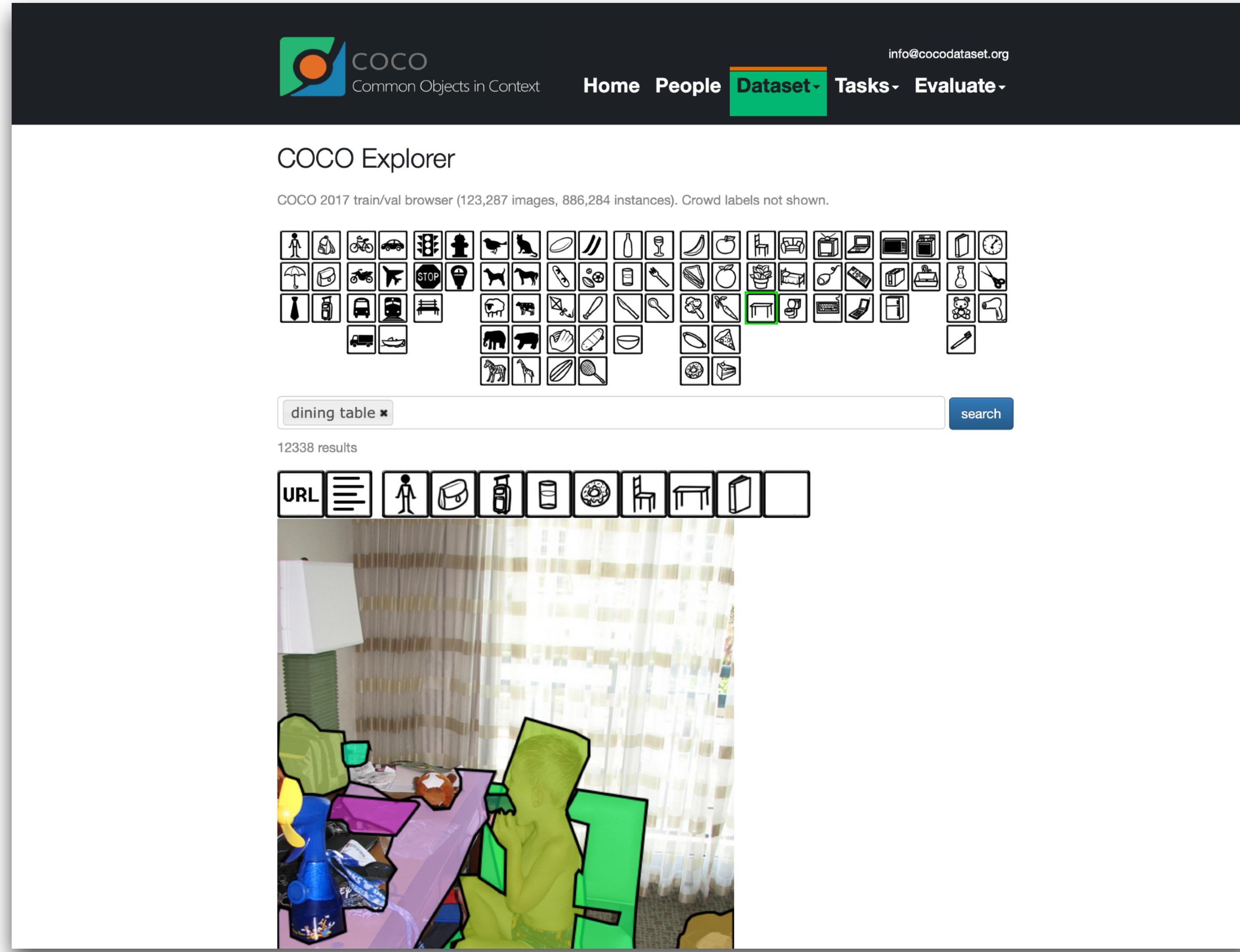
Thing Constellation Visualiser



Data-Driven Design Exploration



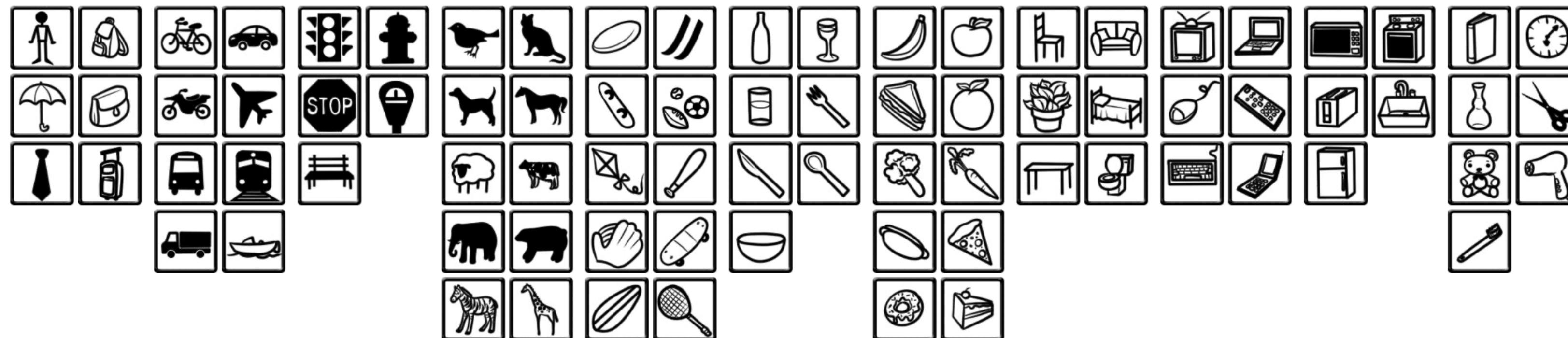
Dataset inside thingCV: COCO dataset



<https://cocodataset.org/#home>

Exercise 1: Design Your AI agents (20 mins)

- Every group/individual use 15 mins to
 - play with an **existing image dataset (i.e., COCO dataset)**
 - use **thingCV tool** to explore the ecosystem of everyday objects
 - **redesign 1 everyday object** from 80 objects, and think about how this object interact with people.



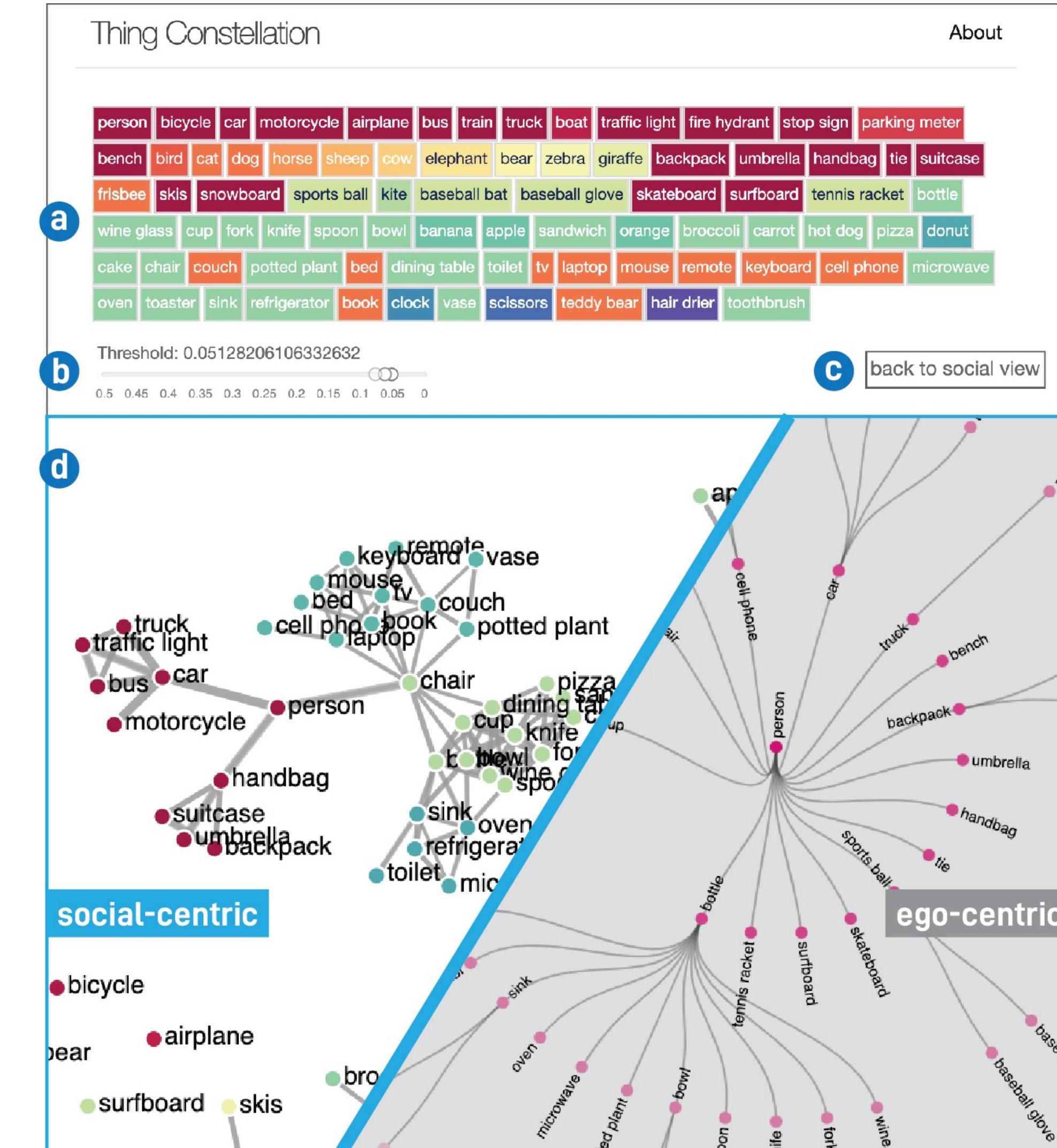
AI Canvas

Concept / Idea Description Describe your idea in 2-3 sentences 1	The Role of Human What specific role does human play in your concept? What specific task does human perform? 3	Input Data Which data items does AI use? How many data does AI need? 5
Contextual situations When and where will the characters (i.e., target users, multiple stakeholder) use your concept? What is the context (i.e., place, environment, time, etc) 2	The Role of AI What specific role does AI play in your concept? What type of AI? What specific task does AI perform? 4	Feature What specific characteristic of the data? A feature is a measurable property of the object (data) you are trying to analyze. Features are independent variables that acknowledge inputs in your system. 6
		Output Results What is the result of your system? Is a binary answer or a numerical number? Or is a multiple-class answer? 7
		Open questions / problems? What AI still cannot do in your concept? Which aspects are still unclear for you in terms of your AI system? 8

Break

Session II: Build your own thingCV

Thing Constellation Visualizer



a Object Panel

- objects in the same community are filled with the same colors.
- click any object to zoom into ego-centric view.

b Threshold Slider

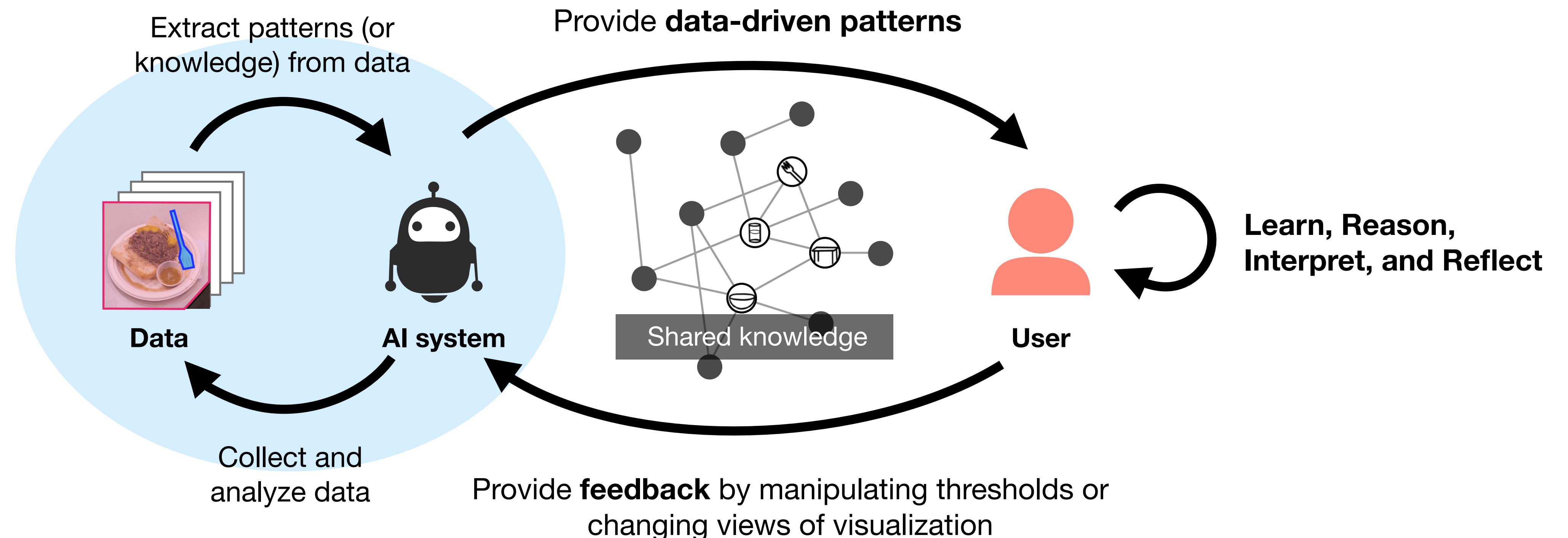
- adjust threshold value to determine the density (i.e. numbers of links between nodes).

c Switch Button

- switch two views under the same threshold value.

d Thing Constellation Visualisation

- based on the previous selections, here shows the result which can be social-centric or ego-centric constellation.



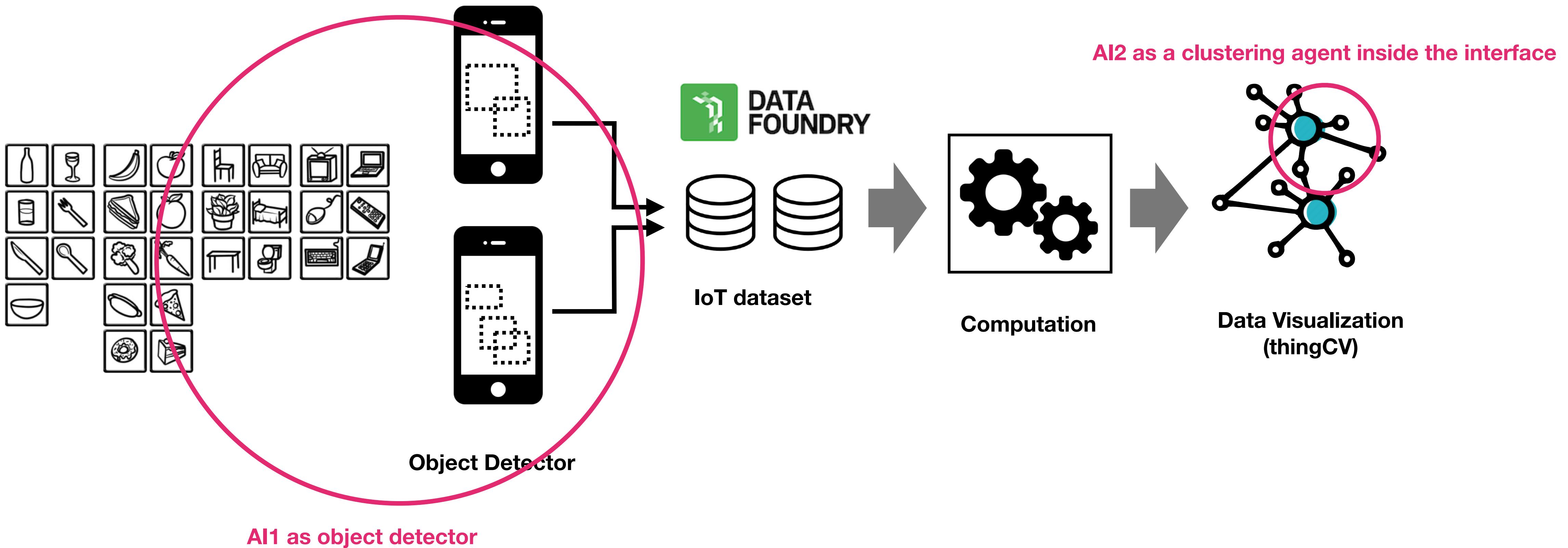
Data-driven pattern + **Human interpretation**

AI Canvas

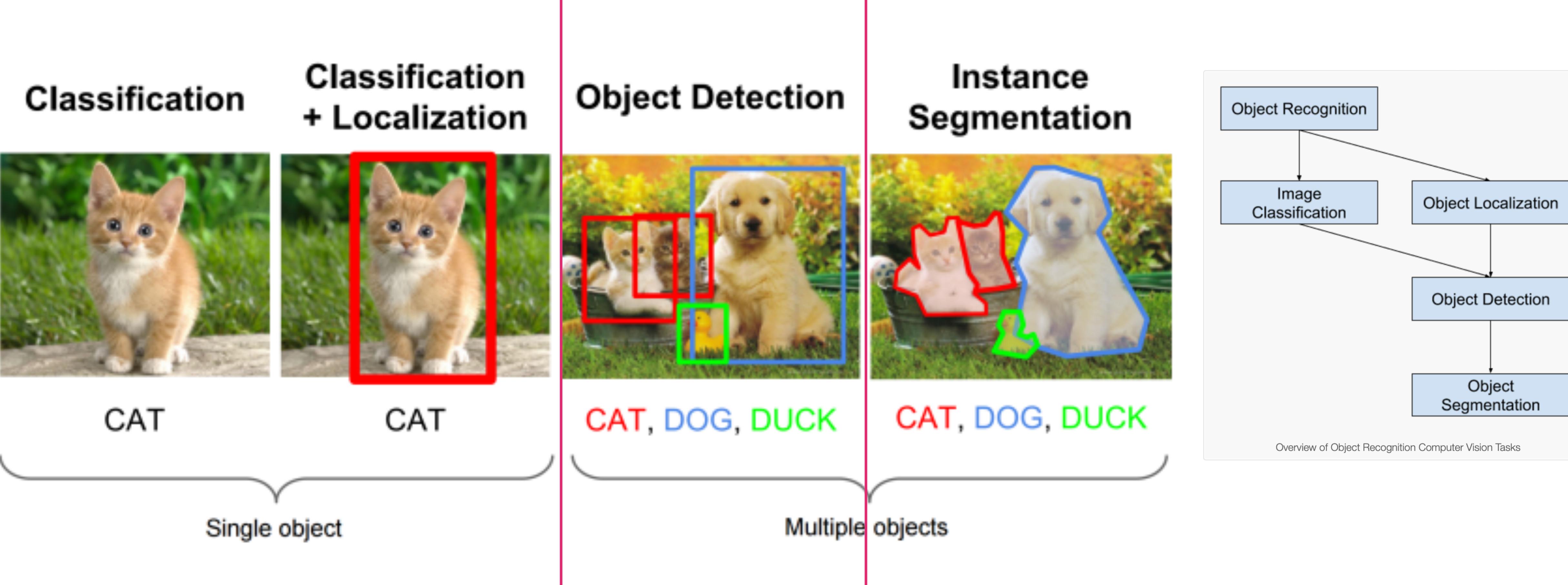
Object detector

<p>Concept / Idea Description</p> <p>Describe your idea in 2-3 sentences</p> <p>Support designers to design IoT ecosystem by enabling them to explore relationships among objects in everyday contexts</p>	<p>The Role of Human</p> <p>What specific role does human play in your concept? What specific task does human perform?</p> <ul style="list-style-type: none">* designers collect data and then fed into AI.* designers interact with data patterns by changing different views through an interface	<p>Input Data</p> <p>Which data items does AI use? How many data does AI need?</p> <ul style="list-style-type: none">* image data collected from phone
<p>Contextual situations</p> <p>When and where will the characters (i.e., target users, multiple stakeholder) use your concept? What is the context (i.e., place, environment, time, etc)</p> <p>Designers use this tool to understand emergent patterns through data captured on everyday contexts, and further use this tool to come up with future scenarios in which objects collaborate with each other</p>	<p>The Role of AI</p> <p>What specific role does AI play in your concept? What type of AI? What specific task does AI perform?</p> <ul style="list-style-type: none">- ThingCV will (1) extracts emergent patterns based on a large numbers of photos, (2) visualize the patterns, (3) enable human to play with data patterns through an interactive interface- Object detector identifies objects from a photo or a camera stream	<p>Feature</p> <p>What specific characteristic of the data? A feature is a measurable property of the object (data) you are trying to analyze. Features are independent variables that act like inputs in your system.</p> <ul style="list-style-type: none">* pixel colors (r,g,b), size of image
		<p>Output Results</p> <p>What is the result of your system? Is a binary answer or a numerical number? Or is a multiple-class answer?</p> <ul style="list-style-type: none">* One or more objects in a phone, each object is indicated by a bounding box, object category, and a confidence score
		<p>Open questions / problems?</p> <p>What AI still cannot do in your concept? Which aspects are still unclear for you in terms of your AI system?</p> <p>8</p>

System Workflow



Object Recognition



Object Detector using ML5.js



```
const video = document.getElementById('video');

// Create a ObjectDetector method
const objectDetector = ml5.objectDetector('cocossd', {}, modelLoaded);

// When the model is loaded
function modelLoaded() {
  console.log('Model Loaded!');
}

// Detect objects in the video element
objectDetector.detect(video, (err, results) => {
  console.log(results); // Will output bounding boxes of detected objects
});
```

p5.js

[Home](#)[Editor](#)[Download](#)[Donate](#)[Get Started](#)[Reference](#)[Libraries](#)[Learn](#)[Teach](#)[Examples](#)[Books](#)[Community](#)[Showcase](#)[Forum](#)

Hello!

Search p5js.org

p5.js is a JavaScript library for creative coding, with a focus on making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else! p5.js is free and open-source because we believe software, and the tools to learn it, should be accessible to everyone.

Using the metaphor of a sketch, p5.js has a full set of drawing functionality. However, you're not limited to your drawing canvas. You can think of your whole browser page as your sketch, including HTML5 objects for text, input, video, webcam, and sound.

[Join the p5.js Discord!](#)

Start creating with the p5 Editor!

Community

We are a community of, and in solidarity with, people from every gender identity and expression, sexual orientation, race, ethnicity, language,



Friendly Machine Learning for the Web

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

What ml5.js can do?

image

ImageClassifier
PoseNet StyleTransfer
BodyPix pix2pix
UNET CVAE
Handpose SketchRNN
Facemesh ObjectDetector
FaceApi

sound

SoundClassification
PitchDetection

text

CharRNN
Sentiment
Word2Vec

helpers

NeuralNetwork
FeatureExtractor
KNNClassifier
Kmeans

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Getting Started with ml5.js</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- p5 -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.0.0/p5.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.0.0/addons/p5.sound.min.js"></script>
    <!-- ml5 -->
    <script src="https://unpkg.com/ml5@latest/dist/ml5.min.js"></script>
  </head>

  <body>
    <script>
      function setup() {
        createCanvas(400, 400);
      }

      function draw() {
        background(200);
      }
    </script>
  </body>
</html>
```

p5*

File ▾ Edit ▾ Sketch ▾ Help ▾

English ▾ Hello, janetyc! ▾

Auto-refresh ImageClassification by ml5

Sketch Files < sketch.js Preview

1 Call ml5 function and load the model

2 Apply the ml5 function to an input (e.g., image, video, text)

3 Do something with the output (e.g., value, labels, points, etc)

Image classification using MobileNet and p5.js



```
8 Image classification using MobileNet and p5.js
9 This example uses a callback pattern to create the classifier
10 === */
11
12 // Initialize the Image Classifier method with MobileNet. A callback needs to be
13 // passed.
13 let classifier;
14
15 // A variable to hold the image we want to classify
16 let img;
17
18 function preload() {
19   classifier = ml5.imageClassifier('MobileNet');
20   img = loadImage('images/bird.jpg');
21 }
22
23 function setup() {
24   createCanvas(400, 400);
25   classifier.classify(img, gotResult);
26   image(img, 0, 0);
27 }
28
29 // A function to run when we get any errors and the results
30 function gotResult(error, results) {
31   // Display error in the console
32   if (error) {
33     console.error(error);
34   }
35   // The results are in an array ordered by confidence.
36   console.log(results);
37   createDiv('Label: ' + results[0].label);
38   createDiv('Confidence: ' + nf(results[0].confidence, 0, 2));
39 }
40
```

Console Clear >

How to use ml5.js?

imageClassifier('MobileNet')



```
const classifier = ml5.imageClassifier('MobileNet');

classifier.classify(video, gotResult);

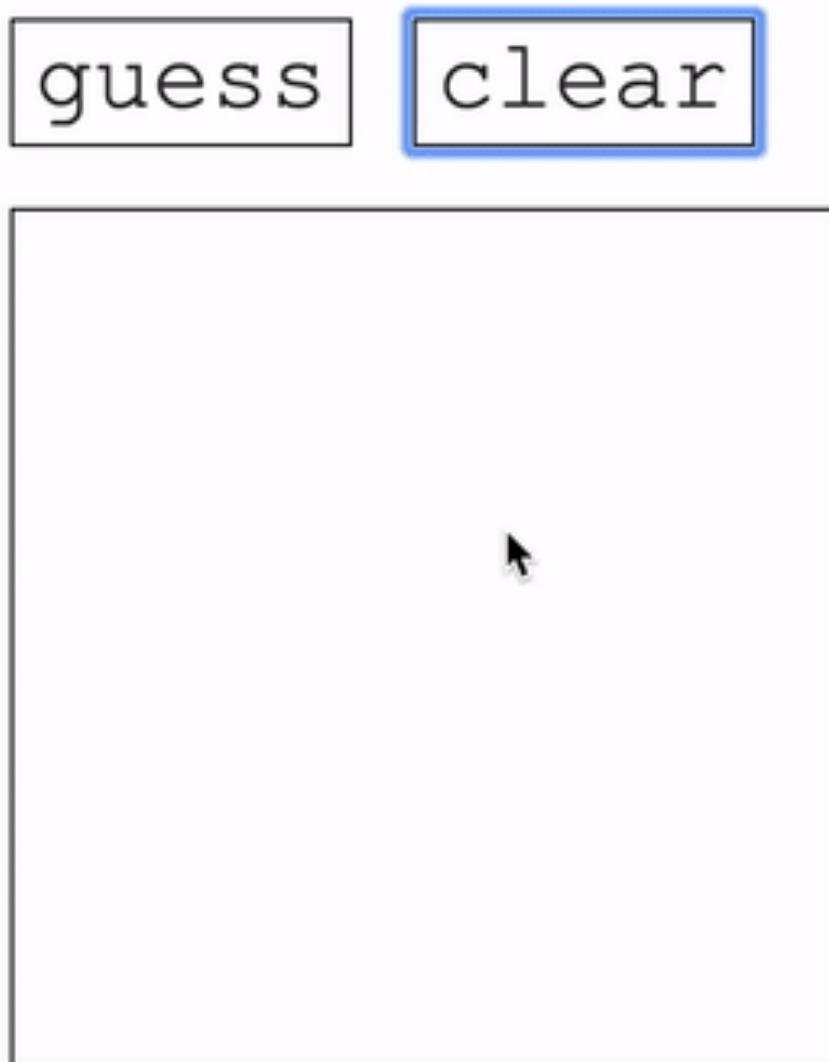
function gotResult(error, result) {
  console.log(result);
}
```

My guess is a toaster.

My confidence is 0.12.

imageClassifier('DoodleNet')

Doodle Classifier on 345 classes



<https://github.com/yining1023/doodleNet>

```
const classifier = ml5.imageClassifier('DoodleNet');

classifier.classify(canvas, gotResult);

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

https://editor.p5js.org/ml5/sketches/ImageClassification_DoodleNet_Canvas

PoseNet



[Image source]

```
const video = document.getElementById('video');

// Create a new poseNet method
const poseNet = ml5.poseNet(video, modelLoaded);

// When the model is loaded
function modelLoaded() {
  console.log('Model Loaded!');
}

// Listen to new 'pose' events
poseNet.on('pose', (results) => {
  poses = results;
});
```

BodyPix

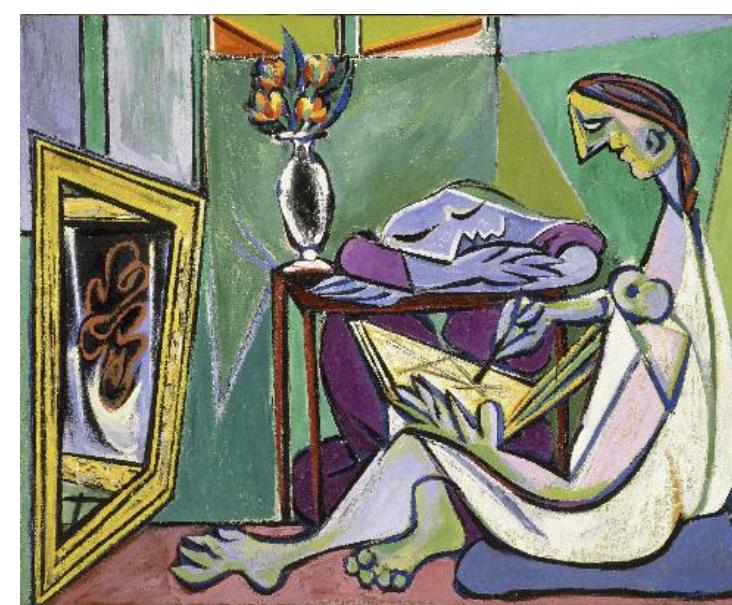


```
const bodypix = ml5.bodyPix(modelReady);

function modelReady() {
  // segment the image given
  bodypix.segment(img, gotResults);
}

function gotResults(error, result) {
  if (error) {
    console.log(error);
    return;
  }
  // log the result
  console.log(result.backgroundMask);
}
```

Style Transfer



[Image source]

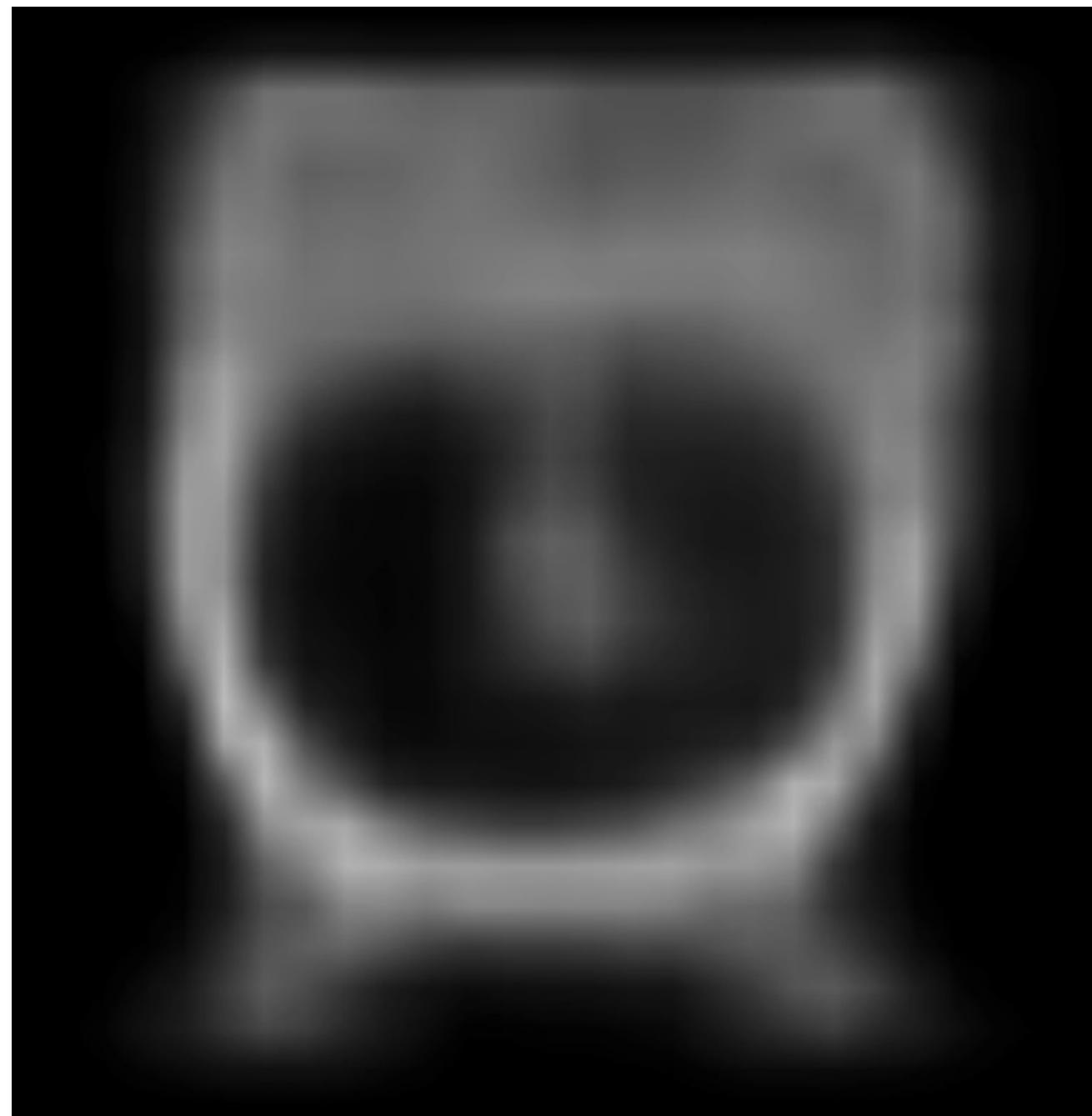
```
// Create a new Style Transfer Instance
const style = ml5.styleTransfer('data/myModel/', modelLoaded);

// When the model is loaded
function modelLoaded() {
  console.log('Model Loaded!');
}

// Grab a img element and generate a new image.
style.transfer(document.getElementById("img"), function(error, result)
{
  img.src = result.src;
});
```

CVAE

(conditional variational autoencoder)



[Image source]

```
const cvae = ml5.CVAE('model/quick_draw/manifest.json', modelReady);

function modelReady() {
  // generate an image of an airplane
  cvae.generate('airplane', gotImage);
}

function gotImage(error, result) {
  if (error) {
    console.log(error);
    return;
  }
  // log the result
  console.log(result);
}
```

CharRNN

This example uses a pre-trained model on a corpus of [Virginia Woolf](#)

seed text:

temperature: 0.5

Model Loaded

The sky was blue and

```
// Create the character level generator with a pre trained model
const rnn = ml5.charRNN('models/bolaño/', modelLoaded);

// When the model is loaded
function modelLoaded() {
  console.log('Model Loaded!');
}

// Generate content
rnn.generate({ seed: 'the meaning of pizza is' }, (err, results) => {
  console.log(results);
});
```

Sentiment Analysis

Sentiment Analysis Demo

This example uses model trained on movie reviews. This model scores the sentiment of text with a value between 0 ("negative") and 1 ("positive"). The movie reviews were truncated to a maximum of 200 words and only the 20,000 most common words in the reviews are used.

model loaded

sentiment score:

```
// Create a new Sentiment method
const sentiment = ml5.sentiment('movieReviews', modelReady);
```

```
// When the model is loaded
function modelReady() {
  // model is ready
  console.log('Model Loaded!');
}
```

```
// make the prediction
const prediction = sentiment.predict(text);
console.log(prediction);
```



**DATA
FOUNDRY**

<https://data.id.tue.nl>

What is Data Foundry?

The screenshot shows the Data Foundry web application interface. On the left is a sidebar with the Data Foundry logo and navigation links: My projects (highlighted in green), Data tools, Documentation, and Support. The main area is titled "Home" and "PROJECTS". A green "ADD PROJECT" button is located in the top right. The page displays a grid of project cards:

- New Data Collecting Project** by Henk Apeldoorn, I-Tang Chiang, Janet Huang, Wietse Loor, Geert van den Boomen. Tags: DIARY, SCRIPT, ENTITY, GOOGLEFIT, FITBIT, EXISTING, IOT.
- [ARTIFICE] AI Workshop 202...** by Janet Huang. Tags: SCRIPT, EXISTING, IOT.
- [DCM210] AI Workshop: Cust...** by I-Tang Chiang, Janet Huang. Tags: SCRIPT, EXISTING, IOT.
- DCB150 Digital Craftsmans...** by Mae-Yin Chan, Janet Huang. Tags: EXISTING.
- Digital Craftsmanship DCB1...** by Stijn Boogaart, van den Janet Huang. Tags: EXISTING.
- [ARTIFICE] AI workshop: Cu...** by I-Tang Chiang, Mathias Funk, Janet Huang. Tags: EXISTING, IOT.
- DCB150 Digital Craftsmans...** by I-Tang Chiang, Janet Huang. Tags: SCRIPT, EXISTING.
- DCB150 DC** by Janet Huang. Tags: EXISTING.
- DCB150 workshop - Chat-G...** by I-Tang Chiang, Janet Huang. Tags: EXISTING, IOT.
- Artifice Workshop** by Janet Huang. Tags: EXISTING.
- Starboard try-out for DBM1...** by I-Tang Chiang, Mathias Funk, Janet Huang. Tags: ENTITY, EXISTING, IOT.
- Artifice demo** by Janet Huang. Tags: EXISTING.
- ARTIFICE Data and AI tools**
- Prototype (P)**
- Prototype (A)**

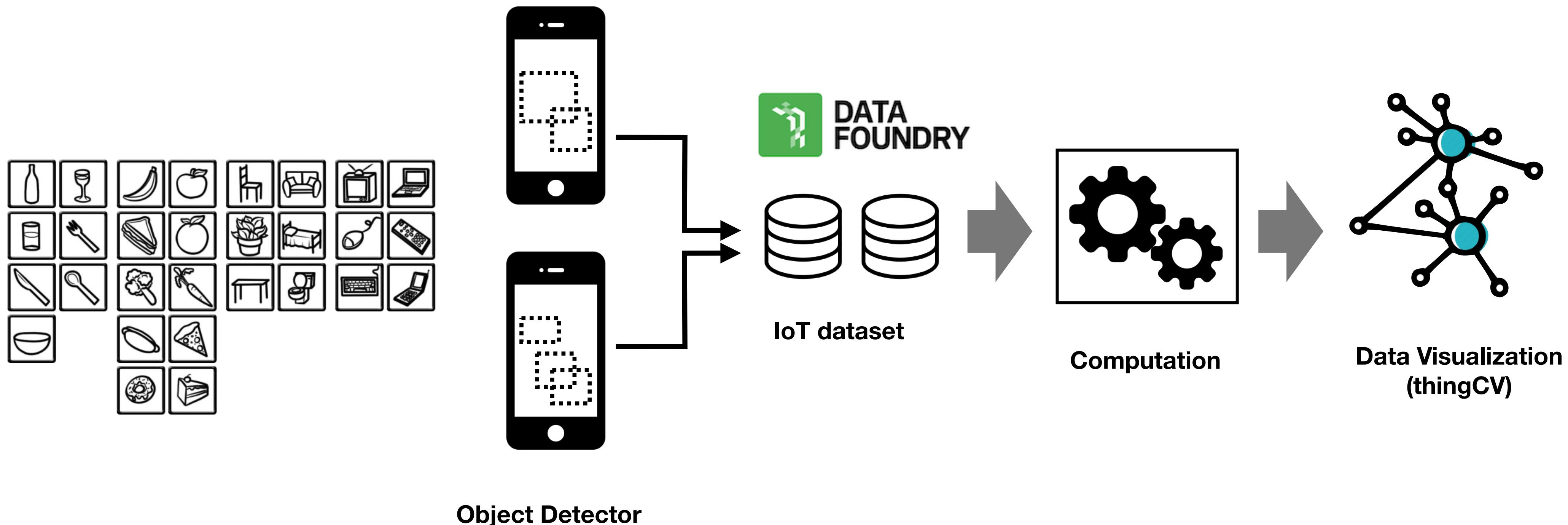
What is Data Foundry?

The screenshot shows the Data Foundry platform's user interface. On the left, a sidebar menu includes 'My projects' (selected), 'Data tools', 'Documentation', and 'Support'. The main area is titled 'PROJECTS' and features a dark header bar with 'Home' and 'ADD PROJECT' buttons. Below this, a grid of project cards is displayed. Each card contains a project title, a brief description, the names of participants, and a list of data sources or types (e.g., DIARY, SCRIPT, ENTITY, GOOGLEFIT, FITBIT, EXISTING, IOT). Some cards also show a 'REDACTED' placeholder. The projects listed include:

- New Data Collecting Project (REDACTED)
- [ARTIFICE] AI Workshop 202... (I-Tang Chiang Janet Huang)
- [DCM210] AI Workshop: Cust... (I-Tang Chiang Janet Huang)
- Digital Craftsmanship DCB150 (Mae-Yin Chan Janet Huang)
- Digital Craftsmanship DCB150 (Stijn Boogaart, van den Janet Huang)
- [ARTIFICE] AI workshop: Cu... (I-Tang Chiang Mathias Funk Janet Huang)
- DCB150 Digital Craftsmanship (I-Tang Chiang Janet Huang)
- DCB150 Digital Craftsmanship (Janet Huang)
- DCB150 workshop (I-Tang Chiang Janet Huang)
- Artifice Workshop (Janet Huang)
- Starboard try-out for DBM1... (I-Tang Chiang Mathias Funk Janet Huang)
- Artifice demo (Janet Huang)
- ARTIFICE Data and AI tools
- Prototype (P)
- Prototype (A)

Session II-(1): Object Detector

System Workflow



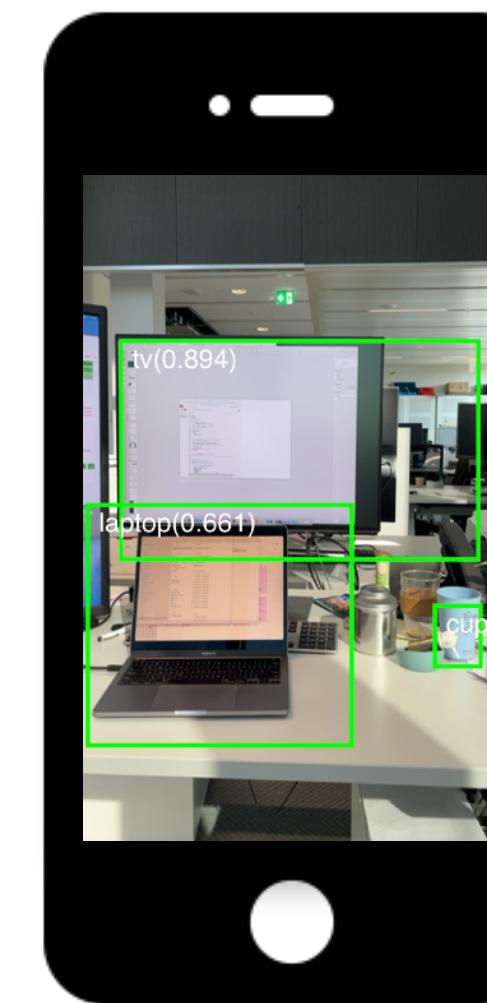
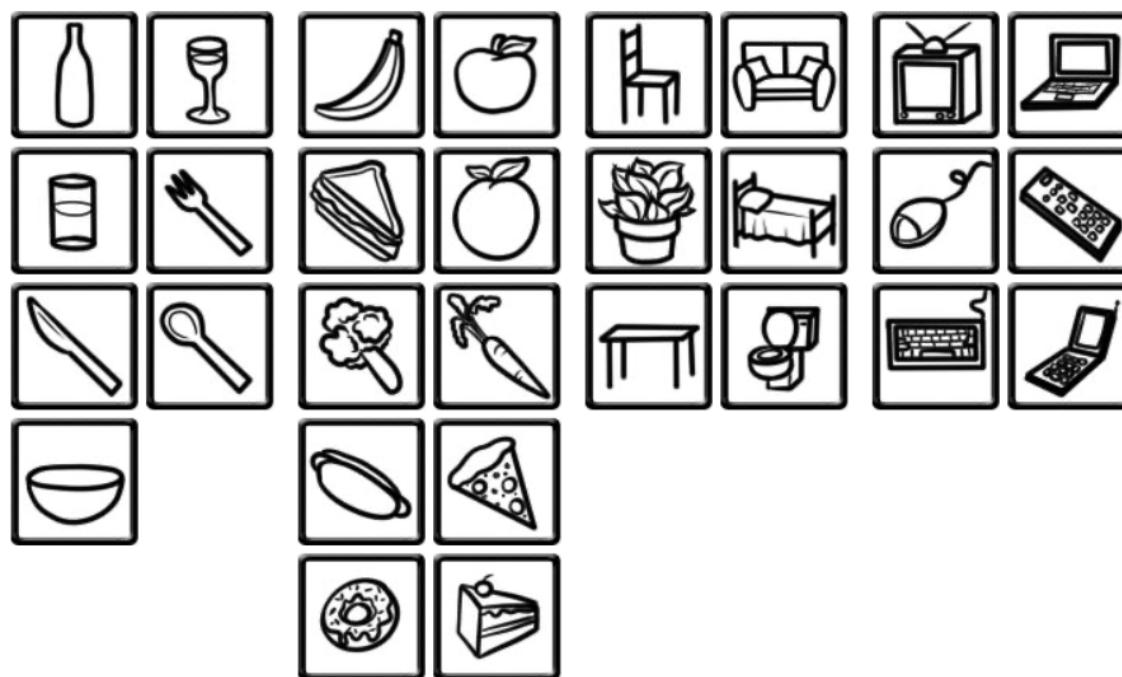


1

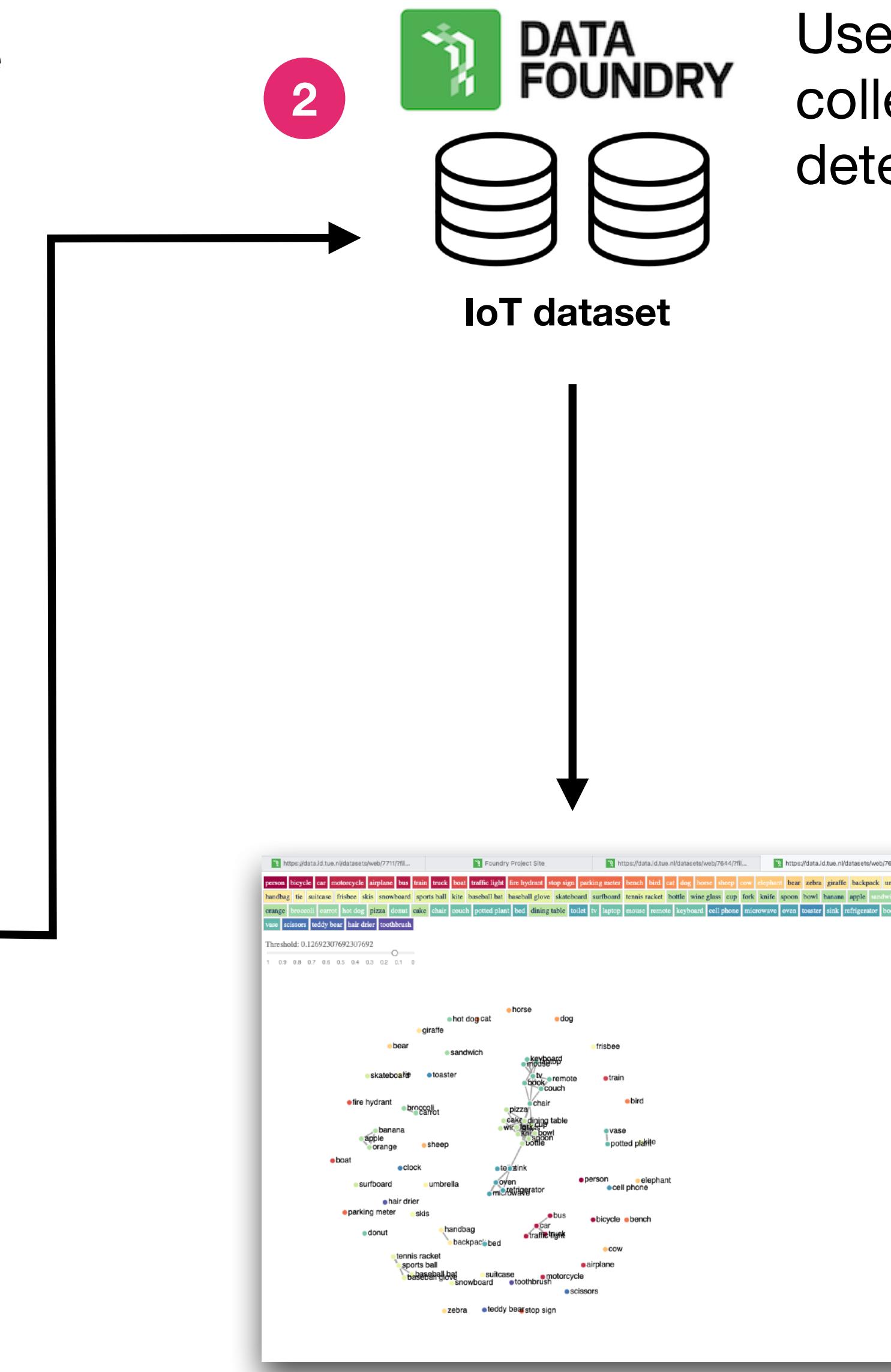
p5.js

ml5

Use DF to host your website
that includes **an object
detector using a pre-trained
model**



Object Detector



Use DF to **store object data**
collected from your object
detector



3

existing dataset

Use DF to host your
website that includes a
data visualization

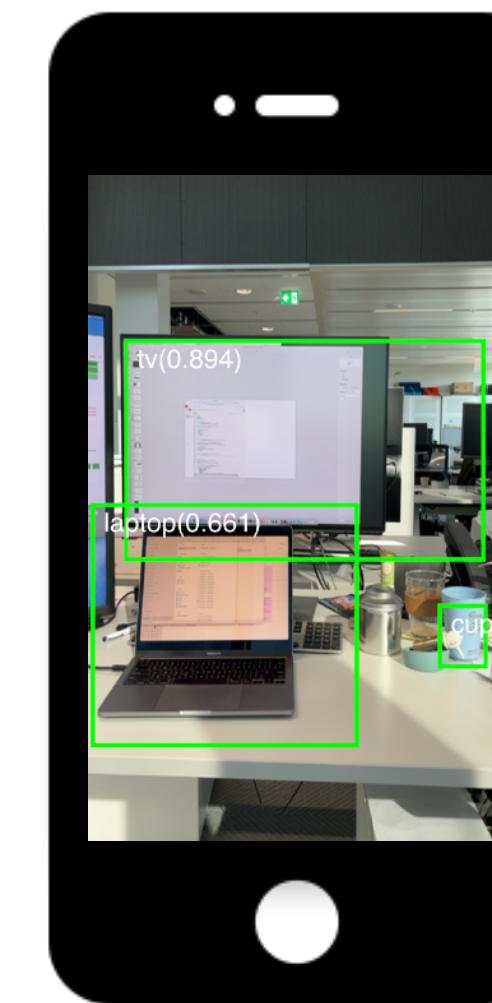
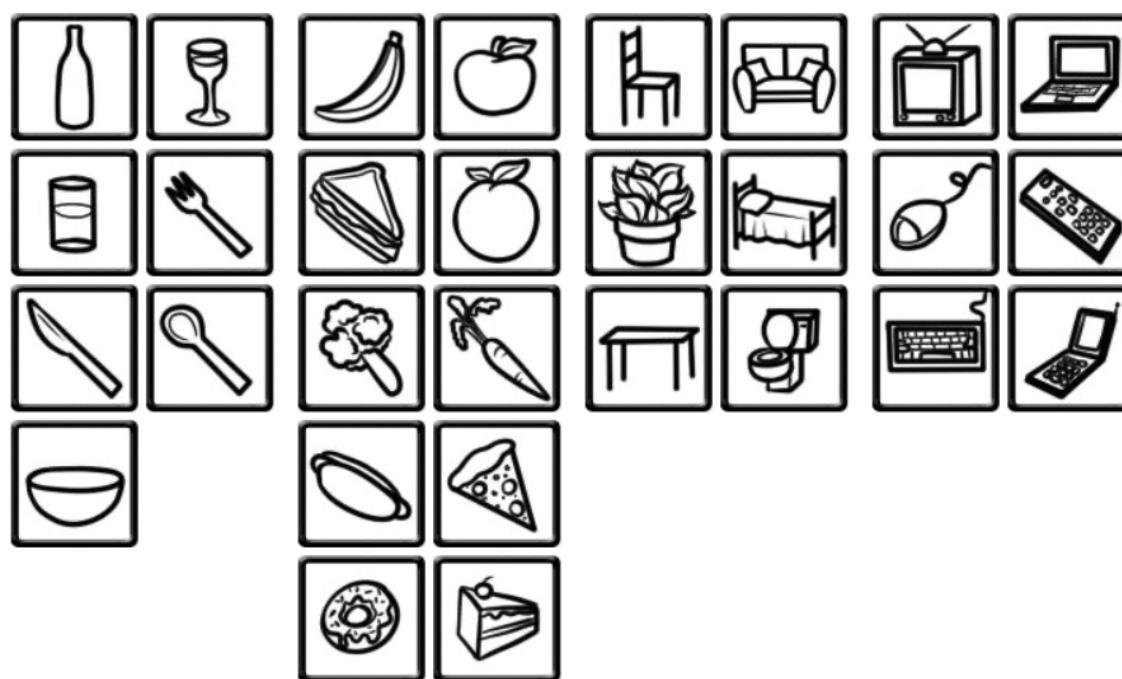


1

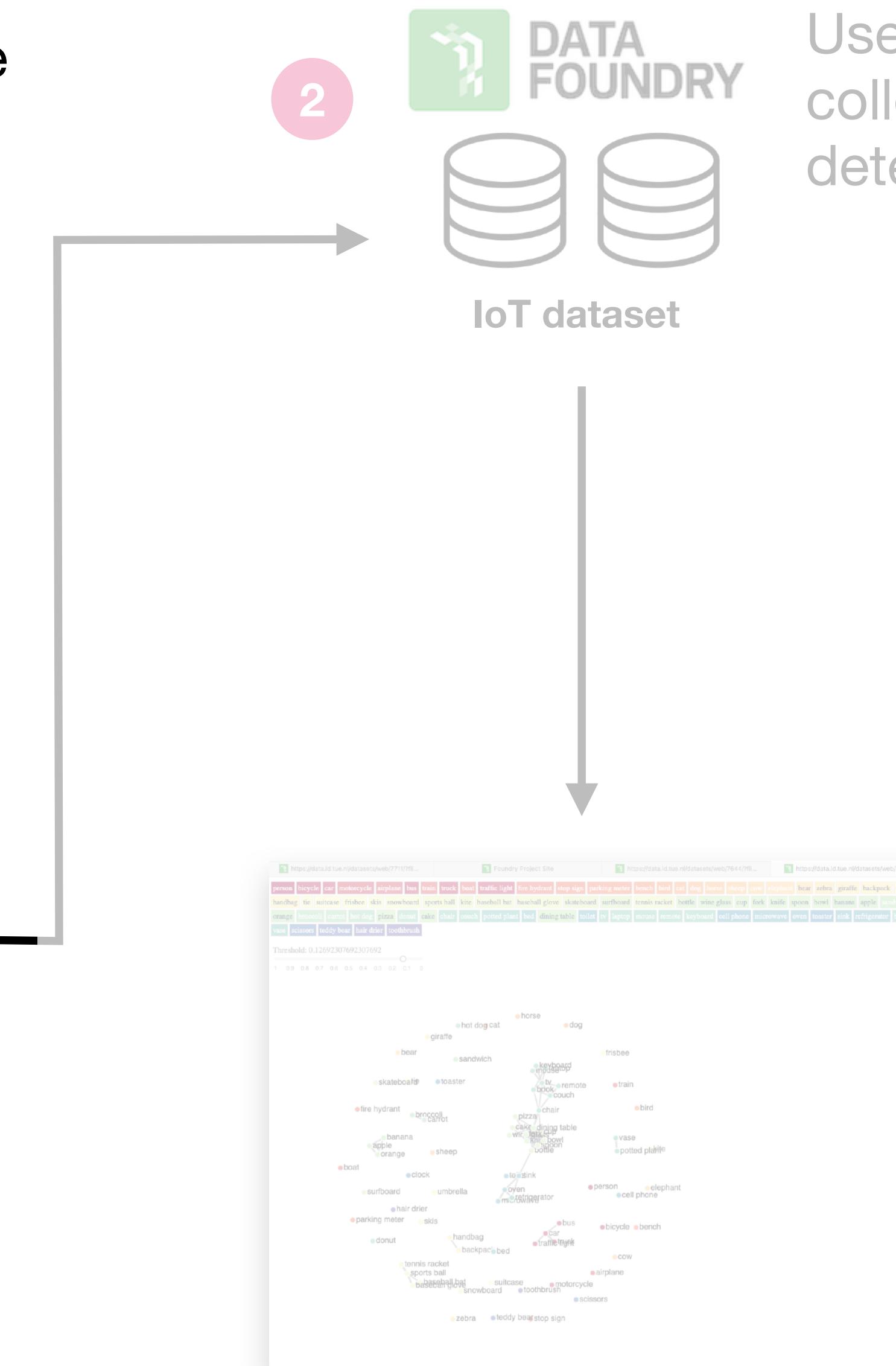
p5.js

ml5

Use DF to host your website
that includes **an object
detector using a pre-trained
model**



Object Detector



Use DF to **store object data**
collected from your object
detector



3

Use DF to host your
website that includes a
data visualization

editor.p5js.org

gmail Research agent wiki Good Sites Class Tools Dict Design tool oTranscribe Time Zone academics - Dropbox TU/e 圖庫&Icon

p5* File Edit Sketch Help English Hello, janetycl!

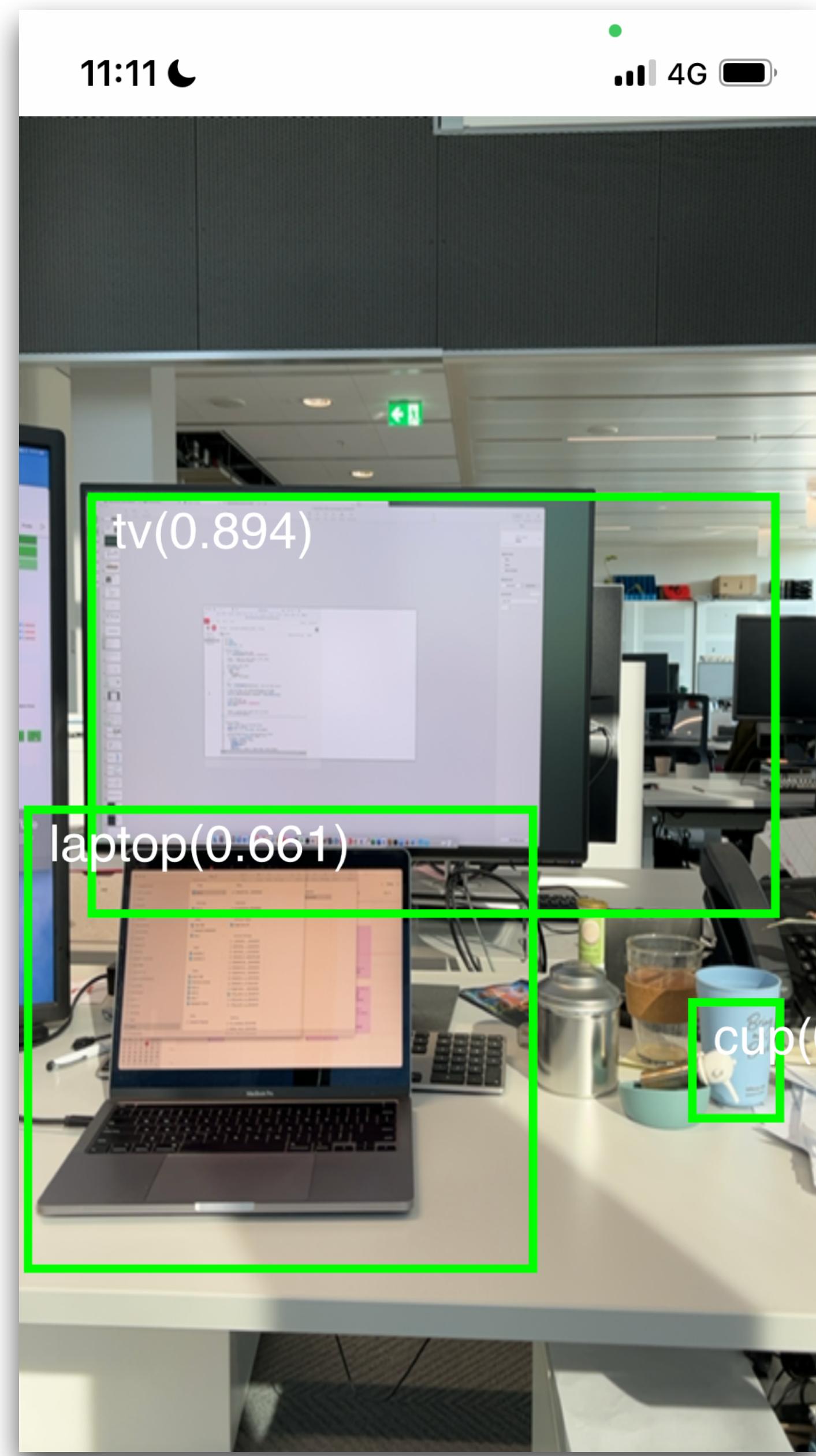
Auto-refresh MDC_Workshop_1_ObjectDetector_Webcam by janetyc

Sketch Files sketch.js

Saved: about 18 hours ago Preview

```
8
9 let cnv;
10 let video;
11 let detector;
12 let detections = [];
13
14 function setup() {
15 // cnv = createCanvas(500, 500);
16 cnv = createCanvas(windowWidth, windowHeight);
17
18 //Step 1: change the front camera to back camera
19 //video = createCapture(VIDEO);
20
21 //use phone's back camera
22 var constraints = {
23 audio: false,
24 video: {
25 facingMode: {
26 exact: "environment"
27 }
28 }
29 };
30 video = createCapture(constraints); //save the video results
31
32 // The line below + the videoLoadedCallback were added
33 // after the video was shot to fix compatibility issues.
34 video.elt.addEventListener('loadeddata', videoLoadedCallback);
35
36 // set video size
37 //video.size(500, 500);
38 video.size(windowWidth, windowHeight);
39 video.hide();
40
41
42 //Step 3: log data when people touch the screen
43 //cnv.touchEnded(logData);
44 }
45
46
47 function draw() {
48 //Step 2: draw video and detected objects
49 //draw video on canvas
50 image(video, 0, 0, video.width, video.height);
51
52 //draw bounding boxes for detected objects on canvas
53 for (let i = 0; i < detections.length; i++) {
54 let object = detections[i];
55 if(object.confidence > 0.6) {
56 stroke(0, 255, 0);
57 strokeWeight(4);
58 noFill();
59 rect(object.x, object.y, object.width, object.height);
}
}
}
}
```

Console Clear



```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Getting Started with ml5.js</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- p5 -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.0.0/p5.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.0.0/addons/p5.sound.min.js"></script>
    <!-- ml5 -->
    <script src="https://unpkg.com/ml5@latest/dist/ml5.min.js"></script>
  </head>
```

```
<body>
  <script>
    function setup() {
      createCanvas(400, 400);
    }

    function draw() {
      background(200);
    }
  </script>
</body>
</html>
```

Or you could put the code into “sketch.js” and include it here

```
<body>
  <script src="sketch.js"></script>
</body>
```

```
9
10 let cnv;
11 let video;
12 let detector;
13 let detections = [];
14
15 function setup() {
16   cnv = createCanvas(windowWidth, windowHeight);
17   cnv.touchEnded(logData);
18   var constraints = {
19     audio: false,
20     video: {
21       facingMode: {
22         exact: "environment"
23       }
24     }
25   };
26   video = createCapture(constraints);
27
28 // The line below + the videoLoadedCallback were added
29 // after the video was shot to fix compatibility issues.
30 video.elt.addEventListener('loadeddata', videoLoadedCallback);
31
32 video.size(windowWidth, windowHeight);
33 video.hide();
34}
35
36
37 function draw() {
38   image(video, 0, 0, video.width, video.height);
39
40   for (let i = 0; i < detections.length; i++) {
41     let object = detections[i];
42     if(object.confidence > 0.6) {
43       stroke(0, 255, 0);
44       strokeWeight(4);
45       noFill();
46       rect(object.x, object.y, object.width, object.height);
47       noStroke();
48       fill(255);
49       textSize(24);
50       text(object.label+(object.confidence.toFixed(3)), object.x + 10, object.y + 24);
51     }
52   }
53 }
```

setup

draw

```
54
55 //solve loaded video issues
56 function videoLoadedCallback() {
57   print("Video Loaded");
58
59 //call model here
60 // Models available are 'cocossd', 'yolo'
61 detector = ml5.objectDetector('cocossd', modelReady);
62 }
63
64 function modelReady() {
65   detector.detect(video, gotDetections);
66 }
67
68 function gotDetections(error, results) {
69   if (error) {
70     console.error(error);
71   }
72   detections = results;
73   detector.detect(video, gotDetections);
74 }
```

objectDetector

Portfolio

My projects

Archive

Community

Collaborations

Subscriptions

Explore

Data tools

Guides

Support

Home

PROJECT STATUS

ADD PROJECT

PROJECTS

MY DIARY

post a diary via telegram bot

Janet Huang

EXISTING

TELEGRAM DIARY

Post a diary via telegram bot

Janet Huang

DIARY MEDIA

THING DIARY

collect object story

Janet Huang

DIARY MEDIA

ARTIFICE WORKSHOP

starboard with ml5.js

Janet Huang

EXISTING

ARTIFICE DEMO

demo

Janet Huang

EXISTING

DCB150 DATA WORKSHOP

data workshop

Janet Huang

EXISTING

[ARTIFICE] AI WORKSHOP: CUS...

Build your own thingCV using object detector

Janet Huang

IOT EXISTING

[ARTIFICE] OBJECT HUNTER

object hunter for 80 objects

Janet Huang

EXISTING

Data Foundry

Data Foundry is a platform that supports researchers to design intelligent interactive product.

It enables an easier way of collecting data, storing data, connecting data, and sharing data with other people. It also provides multiple useful tools for researchers to do rapid prototyping.

Data Foundry: Existing Dataset

[ARTIFICE] AI WORKSHOP 2023: CUSTOMIZE THINGCV

DATA FOUNDRY

- My projects
- Data tools
- Documentation
- Support
- Profile
- Logout

Diary Dataset
Data by participants as diary entries 📝

Media Dataset
Media files (images) 🎥

Existing Dataset
One or more files of an existing dataset (data, text, images, audio, html)

Movement Dataset
Import one or more files of a movement dataset (GPX) 🏃

Experience Sampling Dataset
Import one or more files of an experience sampling dataset (created by PIEL)
EXISTING Web access

ADD DATASET +

RESEARCHERS
Janet Huang

EDIT PROJECT

PROJECT ID: 4197

Public

Creation: 2023-10-01 -

REQUEST

Portfolio
My projects
Archive
Community
Collaborations
Subscriptions
Explore
Data tools
Guides
Support

Home > [DCM210] AI Workshop: Customized your thingCV > Object Detector using ml5.js

id: 7641 EXISTING PUBLIC MIT

OBJECT DETECTOR USING ML5.JS

2023-03-01 2024-03-01

an object detector that can detect 80 objects through a camera

INFO

License: MIT

VIEW DATA
DOWNLOAD
UPLOAD FILE(S)

events

02 AM day

“sketch.js” is the main file we are working on, including init() and draw()

DATASET FILES

File name	Description	Uploaded	Action
sketch.js	object detector	Mar 01 at 14:27	edit delete
style.css	object detector	Mar 01 at 14:27	edit delete
index.html	object detector	Mar 01 at 14:27	edit delete

CONFIGURATION

CSV/JSON TOK... WEB-ACCESS OOCST STREAM

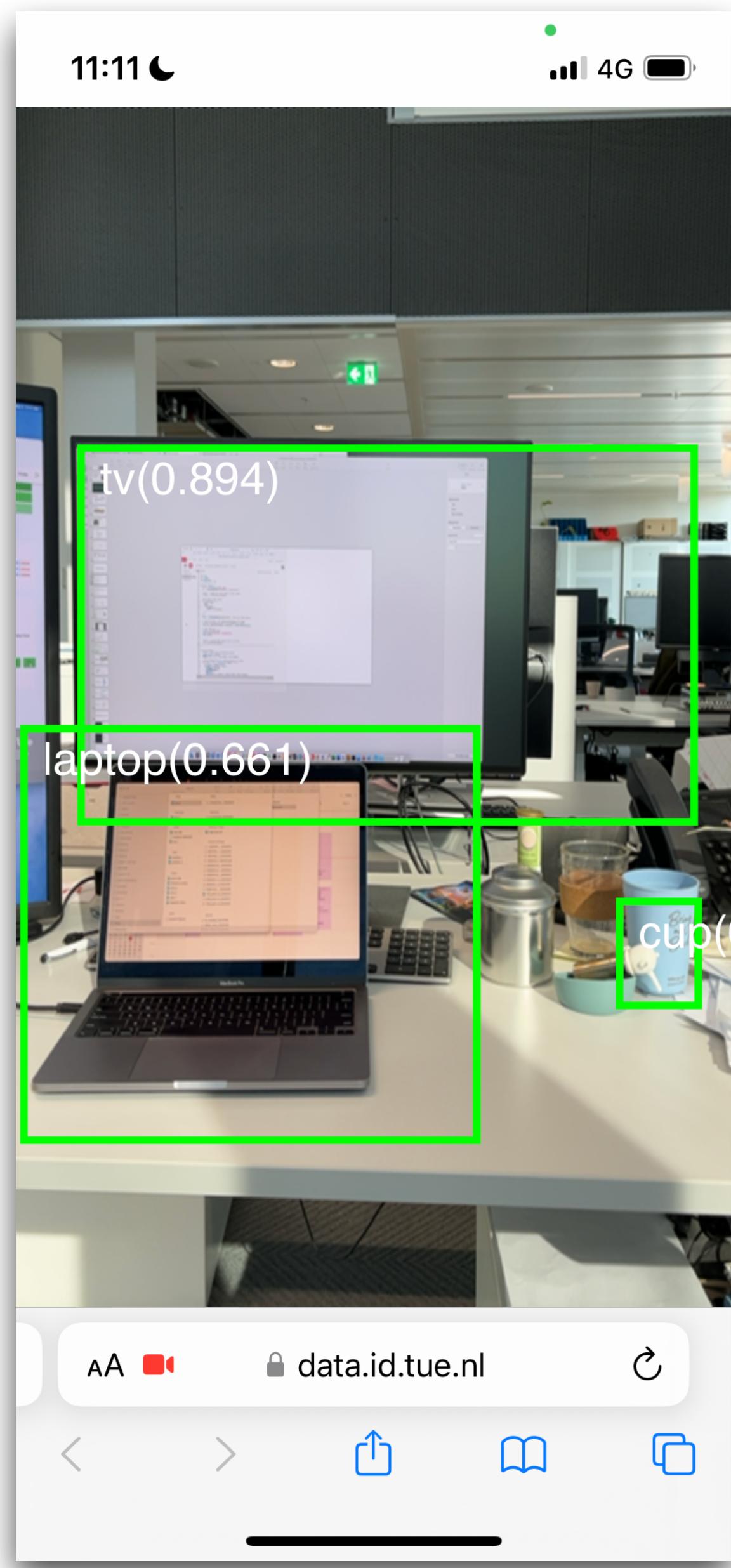
Access the data as a website

As the research team, you can always access the dataset as a [website](#), if the dataset is [active](#). (This is independent from public accessibility.)

This dataset is accessible as a public website with the following link:

<https://data.id.tue.nl/web/dStvaTU4ZGZKVVZLeGtPNU5FTVVlQWs1MFFidk1xM1ZQUVlp>

ACTIVATE WEB ACCESS DISABLE WEB ACCESS



Exercise 2: Build a object detector (20 mins)

Step 1: Create you DF project

Step 2: Create “an existing dataset”

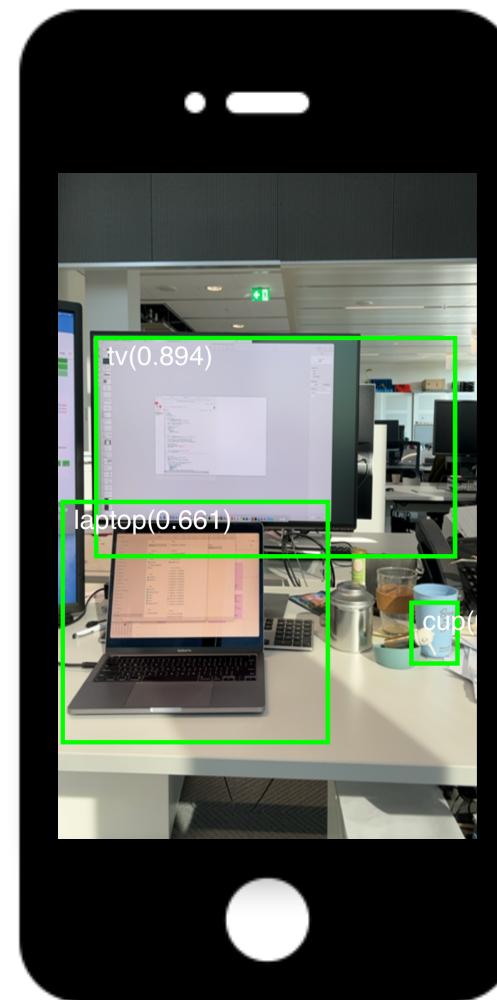
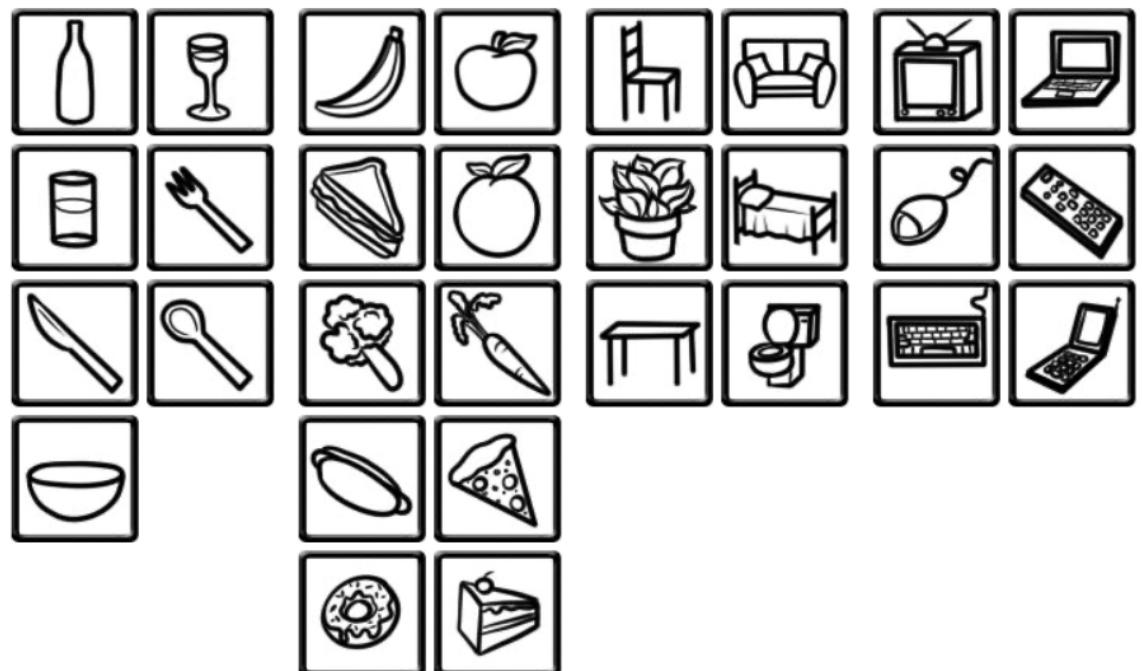
Step 3: Download the source code and submit the code
to the existing dataset

Step 4: Find the link and play with your object detector



Use DF to host your website
that includes **an object
detector using a pre-trained
model**

80 everyday objects



Object Detector

Share your thoughts with your peers!!

Break

Session II-(2) : Store data in DF

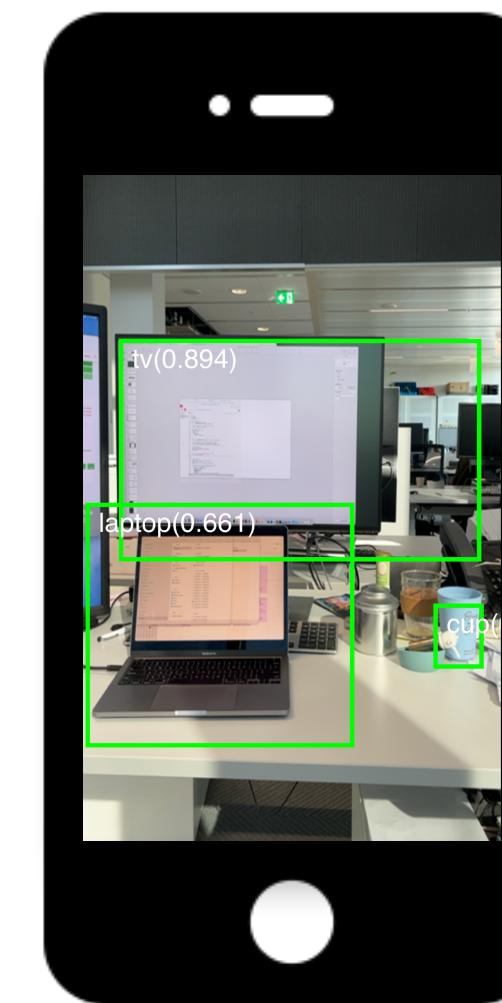
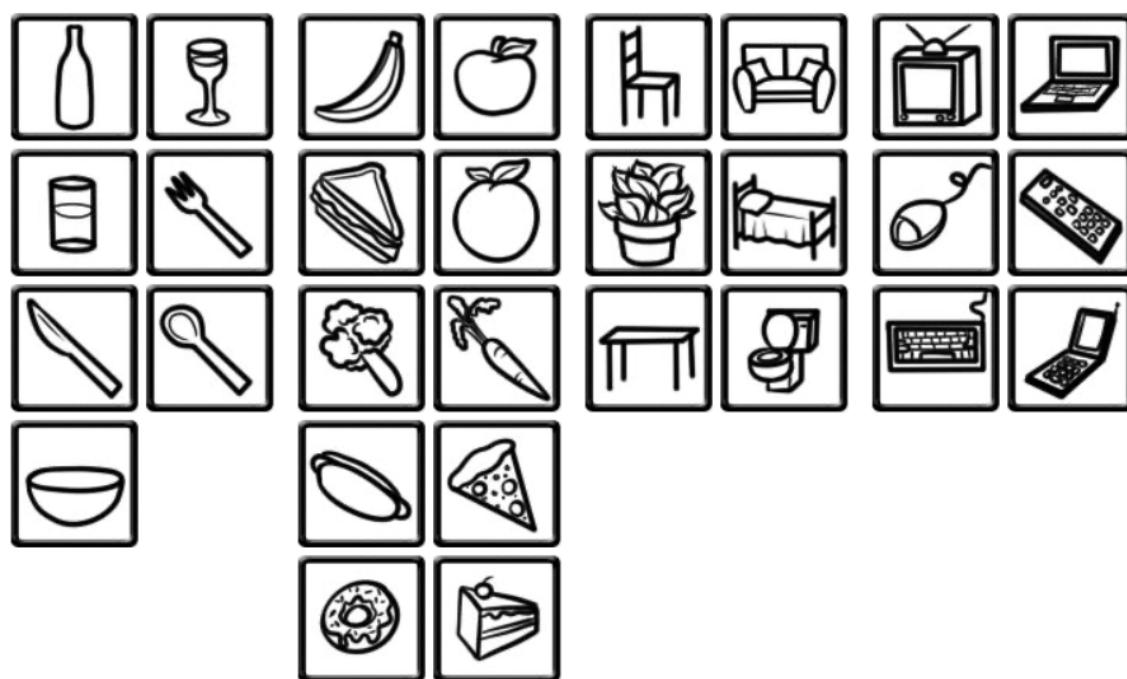


1

p5.js

ml5

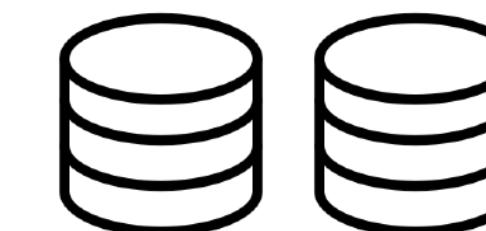
Use DF to host your website
that includes **an object
detector using a pre-trained
model**



Object Detector



2



IoT dataset

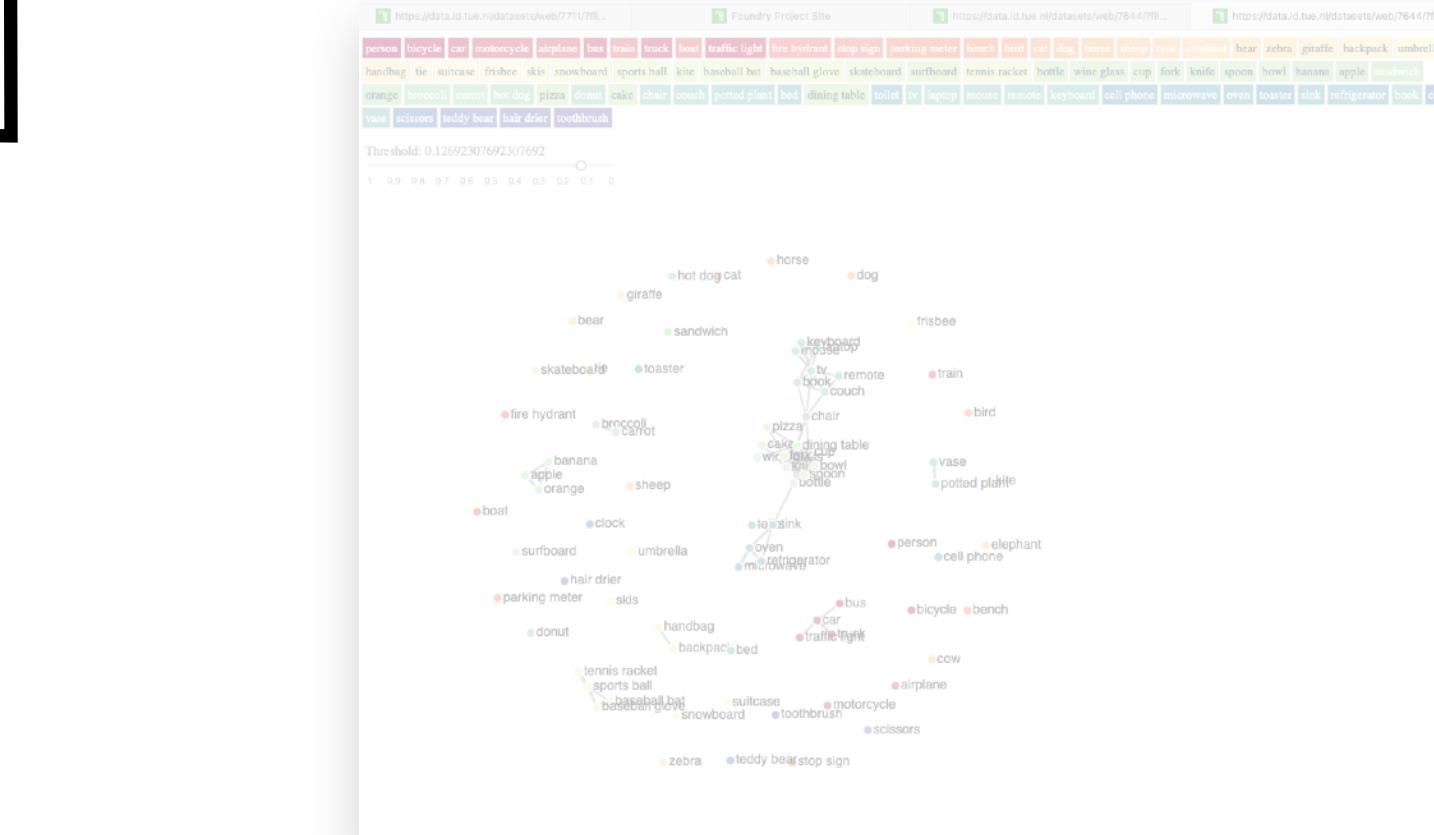
Use DF to **store object data**
collected from your object
detector



3

existing dataset

Use DF to host your
website that includes a
data visualization



Data Foundry: IoT Dataset

The screenshot shows the Data Foundry web application interface. On the left is a sidebar with navigation links: Portfolio, My projects, Archive, Community (which is highlighted in green), Collaborations, Subscriptions, Explore, Data tools, Guides, and Support. The main area has a header with a search icon, profile icon, and a 'Profile' link. Below the header, a message says "Build your own thingCV using object detector". A central modal window titled "CHOOSE A DATASET TYPE" lists several options:

- Script**: Script to automate stuff 🔃
- FitBit Dataset**: Data from connected FitBit devices 🏃
- GoogleFit Dataset**: Data from connected GoogleFit devices 🏃
- IOT Dataset**: Data from connected devices 🌐
- Entity Dataset**: JSON Database for variable data 🏛
- Form Dataset**: Data is collected by a simple form 📝
- Annotation data set**: Data by researcher as annotations 🧩
- Diary Dataset**: Data by participants as diary entries 📇

CONFIGURATION

HTTP-POST

OCSI STREAM

CSV/JSON TOKEN LINK

OCSI STREAM

HTTP POST requests for this dataset (ID: 9330)

POST requests are easy to send from a wide variety of platforms and technologies. Find a list of possible options on the right side.

To allow these requests to store data in the dataset, you need to send them with a special token:

Token

N3pCQTRjbUlIdFdWMXZkTmJpOFBIMmlCOGICRnhPUGRhTd1amt2TnhDUT0=

GENERATE DELETE

Generate a token to activate this inlet, delete the token to deactivate.

HTTP POST Diagnostics



How to use this?

JavaScript

```
var data = { ... your data goes here ... }
var jsonBody = {
  activity: 'ACTIVITY',
  source_id: 'DEVICE_ID',
  data: JSON.stringify(data)
}
fetch('https://data.id.tue.nl/datasets/ts/record/9330/N3pCQTRjbUlIdFdWMXZkTmJpOFBIMmlCOGICRnhPUGRhTd1amt2T'
  method: 'POST',
  mode: 'cors',
  cache: 'no-cache',
  headers: {
    'Content-Type': 'application/json'
  },
  redirect: 'follow',
  referrerPolicy: 'no-referrer',
  body: JSON.stringify(jsonBody)
);
```

Replace DEVICE_ID by the refId attribute of any device in the project. Use any ACTIVITY or leave empty. Provide data in JSON format, such as
`{"parameter1": 2, "parameter2": 5}`

```

92
93 function logData() {
94   final_detections = detections
95   objectList = getDetectionObjects(final_detections);
96
97   let data= {
98     time: +(new Date),
99     detections: objectList
100   }
101
102   let jsonBody = {
103     activity: 'Customize_ThingCV',
104     data: JSON.stringify(data)
105   }
106
107   fetch('https://data.id.tue.nl/datasets/ts/record/7642/MzIySFJHSFNqUjkzMHVWSmovSS9DMEhkMitEY25GV3'
108     method: 'POST',
109     mode: 'cors',
110     cache: 'no-cache',
111     headers: {
112       'Content-Type': 'application/json'
113     },
114     redirect: 'follow',
115     referrerPolicy: 'no-referrer',
116     body: JSON.stringify(jsonBody)
117   );
118
119 }
```

data.id.tue.nl

gmail Research agent wiki Good Sites Class Tools Dict Design tool oTranscribe Time Zone academics - Dropbox TU/e 圖庫&Icon

Dataset table view

Home > [ARTIFICE] AI Workshop 2023: Customize ThingCV > Object data > Data table

My projects

Data tools

Documentation

Show entries

Support

Check whether your data is recorded successfully

Search:

OBJECT DATA

id	ts	pp1	pp2	pp3	detections	time
1	2023-10-02T16:56:03				laptop, tv, tv	1696258563822
2	2023-10-02T16:56:04				laptop, tv, cup, tv	1696258564523
3	2023-10-02T16:56:04				laptop, tv, tv	1696258564827
4	2023-10-02T16:56:05				laptop, tv, tv	1696258565008
5	2023-10-02T16:56:05				laptop, tv, tv	1696258565191
6	2023-10-02T16:56:05				laptop, tv, tv	1696258565374
7	2023-10-02T16:56:05				laptop, tv, tv	1696258565517
8	2023-10-02T16:56:10				tv	1696258570971
9	2023-10-02T16:56:11				tv	1696258571168
10	2023-10-02T16:56:11				tv	1696258571399
11	2023-10-02T16:56:11				tv	1696258571571

Profile

Logout

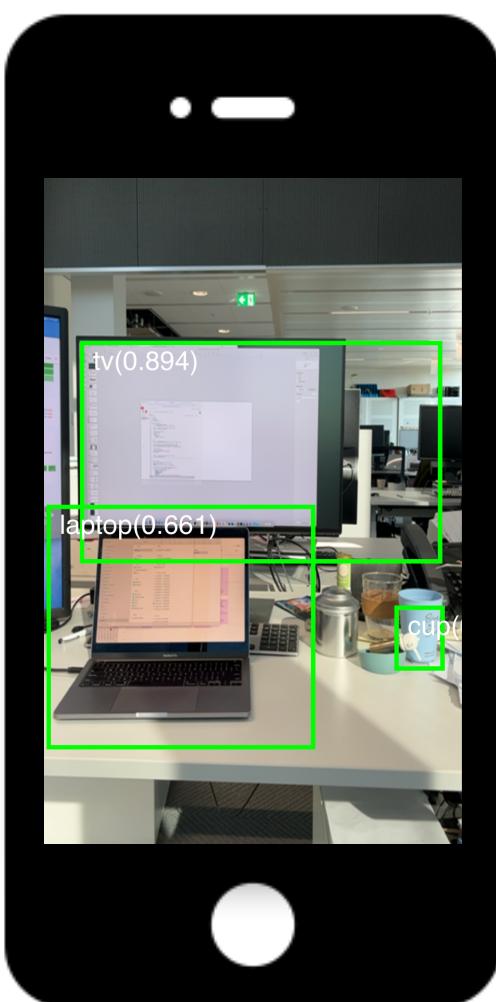
Exercise 3: Store object data to DF

Step 1: Create an IoT dataset

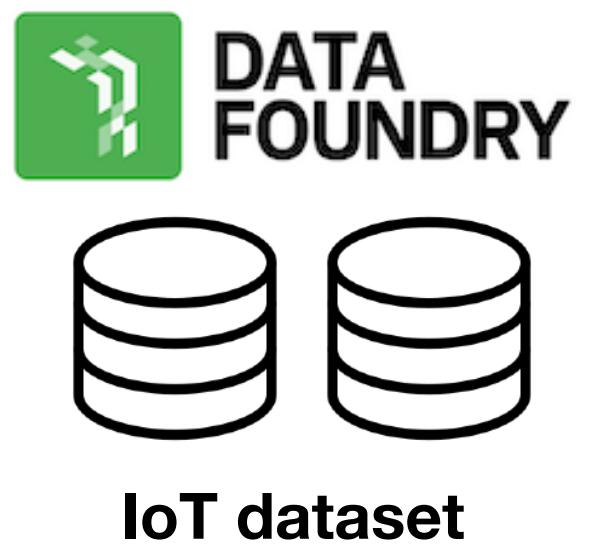
Step 2: Find the HTTP-POST link under the IoT dataset storing your object data

Step 3: Create a mouse-clicked event to upload your object data through the HTTP-POST link

Step 4: Check whether your data been stored on DF successfully



Object Detector



Use DF to **store object data** collected from your object detector



1

p5.js

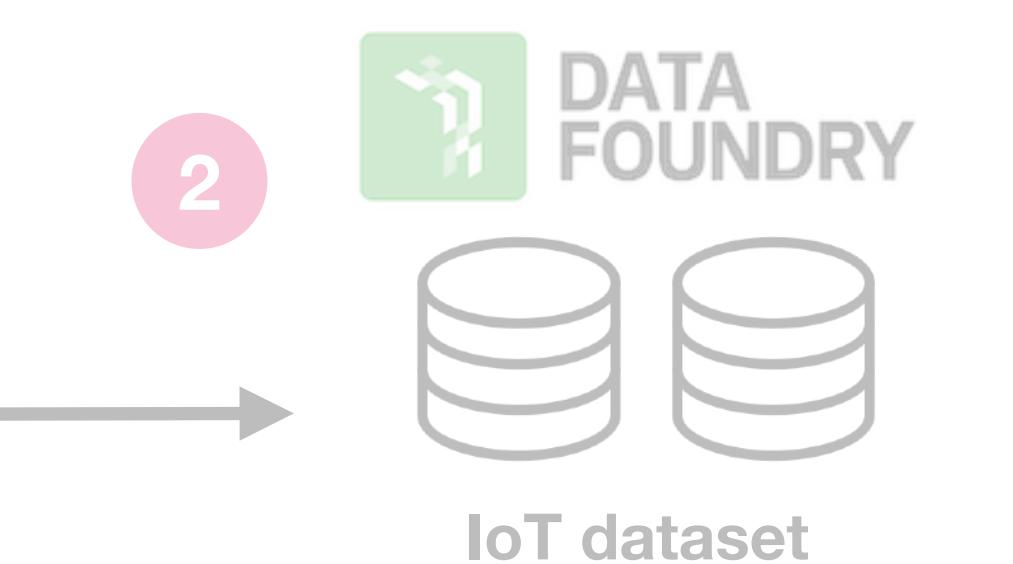
ml5

80 everyday objects



Object Detector

Use DF to host your website
that includes **an object
detector using a pre-trained
model**



Use DF to **store object data**
collected from your object
detector



3

existing dataset

Use DF to host your
website that includes a
data visualization

Session II-(3) : Data Visualizer

Data Foundry: Existing Dataset

[ARTIFICE] AI WORKSHOP 2023: CUSTOMIZE THINGCV

DATA FOUNDRY

- My projects
- Data tools
- Documentation
- Support
- Profile
- Logout

Diary Dataset
Data by participants as diary entries 📝

Media Dataset
Media files (images) 🎥

Existing Dataset
One or more files of an existing dataset (data, text, images, audio, html)

Movement Dataset
Import one or more files of a movement dataset (GPX) 🏃

Experience Sampling Dataset
Import one or more files of an experience sampling dataset (created by PIEL)

EXISTING Web access

ADD DATASET
choose a dataset to add +

RESEARCHERS
Janet Huang

EDIT PROJECT

PROJECT ID: 4197

Public

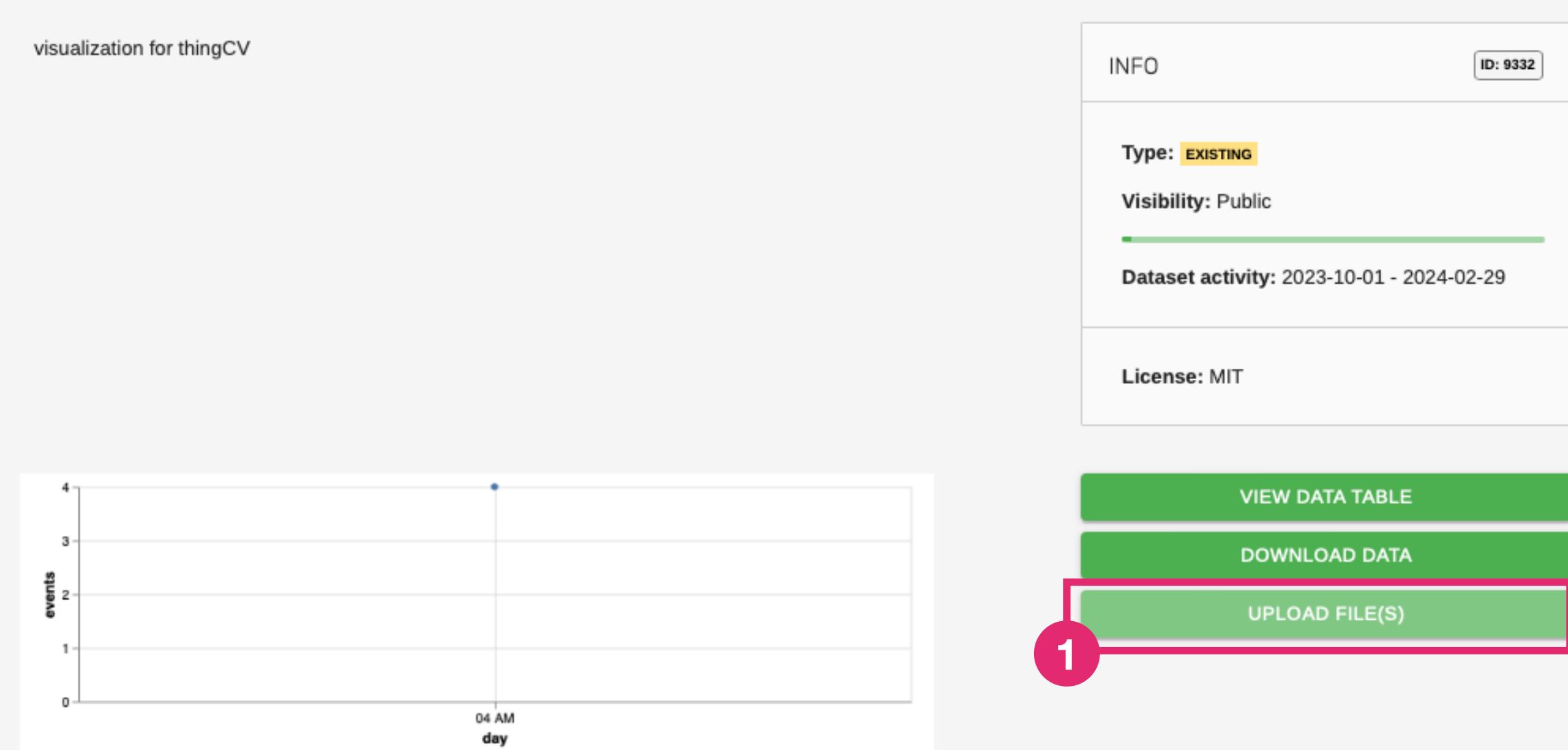
Creation: 2023-10-01 -

REQUEST

THINGCV

visualization for thingCV

EDIT DATASET



2 DATASET FILES

File name	Description	Uploaded	Actions
object_graph.json (open in tab)	thingCV	Oct 02 at 16:58	download edit delete
jLouvain.js (open in tab)	thingCV	Oct 02 at 16:58	download edit delete
jsnetworkx.js (open in tab)	thingCV	Oct 02 at 16:58	download edit delete
index.html (open in tab)	thingCV	Oct 02 at 16:58	download edit delete

CONFIGURATION 3

CSV/JSON TOKEN LINK

WEB-ACCESS 4

OOCST STREAM

Access the data as a website

As the research team, you can always access the dataset as a [website](#), if the dataset is [active](#). (This is independent from public accessibility.)

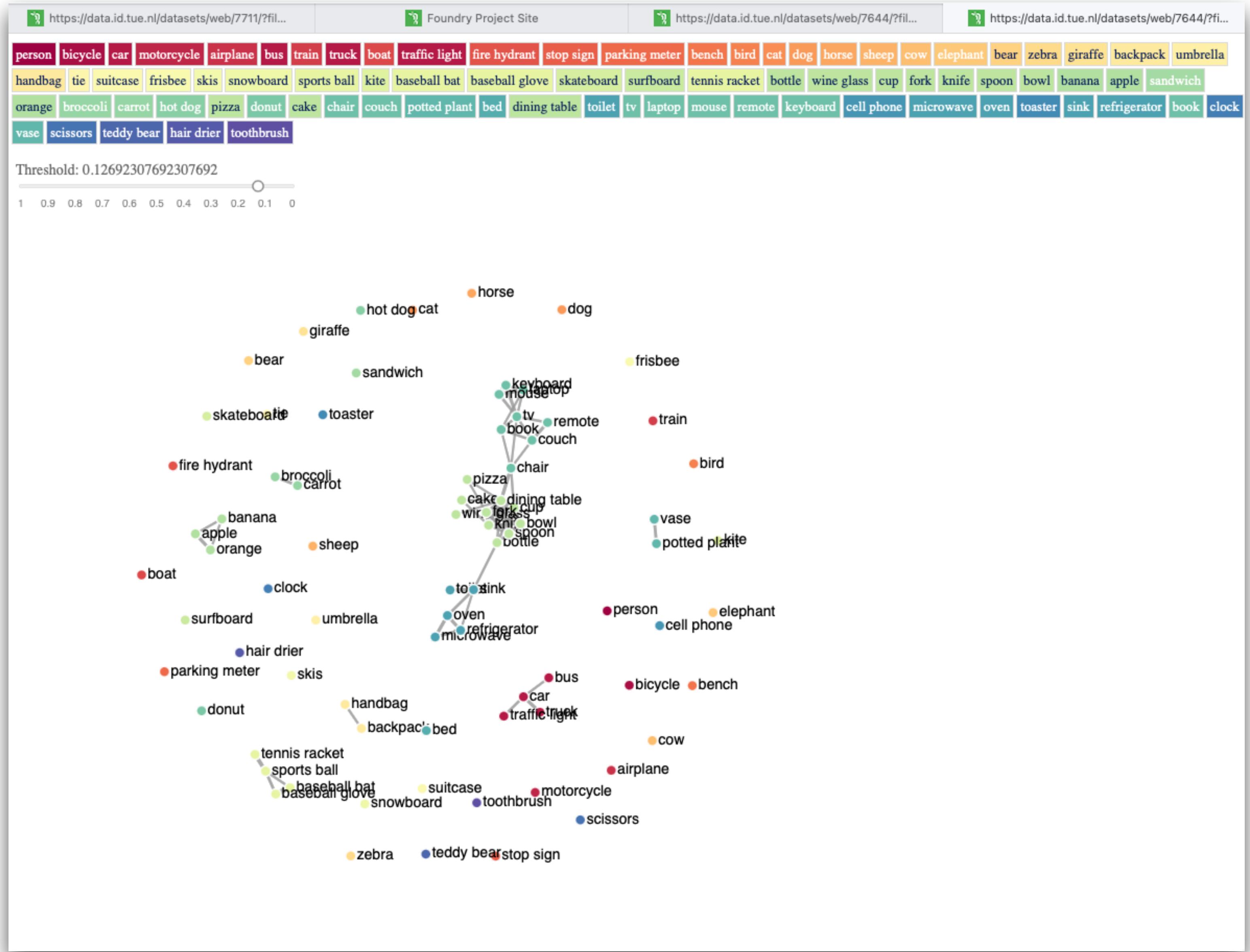
This dataset is accessible as a public website with the following link:

[5](https://data.id.tue.nl/web/elZBQWxRd1F5UDN3N1ExRHkY0VqZ1JCV2FCVThUbG1QZ2htRwqTm9uTT0=)

ACTIVATE WEB ACCESS

DISABLE WEB ACCESS

This screenshot shows the 'CONFIGURATION' section of the dataset page. It includes links for CSV/JSON TOKEN LINK, WEB-ACCESS (which is highlighted with a red box and has a red circle with '4' over it), and OOCST STREAM. Below this is a section for accessing the data as a website, which includes a note about dataset activity and a sharing link. A red circle with '5' highlights the sharing link. At the bottom are 'ACTIVATE WEB ACCESS' and 'DISABLE WEB ACCESS' buttons.



Data visualizer

- use DF existing dataset
- input data

[object_graph.json \(open in tab\)](#)

- source code

[jLouvain.js \(open in tab\)](#)

[jsnetworkx.js \(open in tab\)](#)

[index.html \(open in tab\)](#)

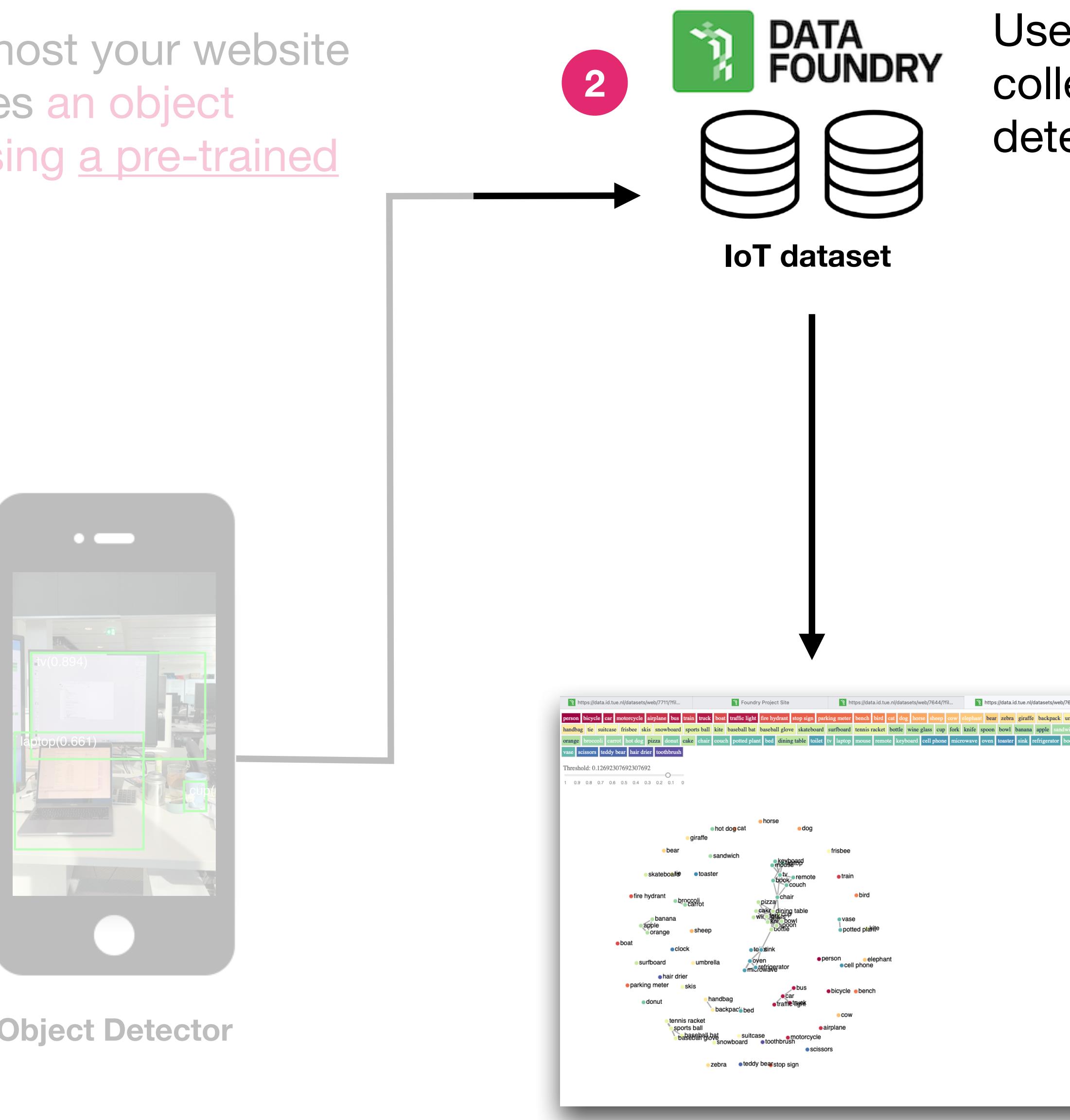


80 everyday objects



Object Detector

Use DF to host your website
that includes **an object**
detector using a pre-trained
model



Use DF to store object data
collected from your object
detector



existing dataset

3

Use DF to host your website that includes a data visualization

Make a connection between IoT dataset and web interface

Home > [ARTIFICE] AI Workshop 2023: Customize ThingCV > Object data

~ OBJECT DATA

EDIT DATASET

CONFIGURATION

1 click CSV/JSON tab

Download and live-access the data as a CSV or JSON file (public link)

You can download the dataset [directly](#) anytime, if you are logged in. If you want to import the data in external software, you can generate a [token](#) link that is publicly accessible. **Be careful: anyone with the link will have access to the data, even outside TU/e. Please ensure that there is no personal data contained in this dataset and that you comply with the statements that you have made in your informed consent form.** To invalidate or revoke a link, just delete it or generate a new one below:

CSV download ([click to download](#))

<https://data.id.tue.nl/datasets/downloadPublic/ZnU4YTBXVzc3UDBPVEw1NW5GdHVkcmQzYzByeW1ldUR3UERTclpsbh>

3 copy JSON link

JSON download ([click to download](#))

<https://data.id.tue.nl/datasets/downloadPublic/json/ZnU4YTBXVzc3UDBPVEw1NW5GdHVkcmQzYzByeW1ldUR3UERTclpsbh>

2 click the button to generate token link

GENERATE TOKEN LINK DELETE

For a mobile dashboard view, use [link](#) or click here:

How to use this?

Python

Home > [ARTIFICE] AI Workshop 2023: Customize ThingCV > thingCV > Edit file

EDIT FILE

Edit contents of 'index.html' below, press 'SAVE' to save your changes.

```
220 //here you need to think of how to get data from the dataset via API
221 var readDataFromDF = async function() {
222   data_url = "https://data.id.tue.nl/datasets/downloadPublic/json/ZnU4YTBXVzc3UDBPVEw1NW5GdHVkcmQzYz
223   var jdata = await fetch(data_url);
224
225   var jsonData = await jdata.json();
226   var object_data = [];
227
228   for(d in jsonData){
229     if(jsonData[d].detections != ""){
230       object_data.push(jsonData[d].detections.split(","));
231     }
232   }
233
234   graph = calculateGraph(object_data);
235   return graph;
236 };
237
238 function calculateGraph(data){
239   obj_img_map = {}
240   data.forEach((obj_list, indx) => {
241     obj_list.forEach(obj => {
242       if(obj in obj_img_map){
243         obj_img_map[obj].push(indx)
244       }else{
245         obj_img_map[obj] = []
246         obj_img_map[obj].push(indx)
247       }
248     })
249   })
250
251   return graph;
252 }
```

4 paste the JSON link here

Change input from reading a local JSON file to reading data from DF

Home > [ARTIFICE] AI Workshop 2023: Customize ThingCV > thingCV > Edit file

EDIT FILE

Edit contents of 'index.html' below, press 'SAVE' to save your changes.

```
300  
301  
302  
303 //read data from DF  
304 var DFdata = readDataFromDF();  
305 DFdata.then(function(graph) {  
306  
307 //read data from json file  
308 //d3.json("own_graph.json", function(error, graph) {  
309 // if (error) throw error;  
310  
311 function getByValue(map, searchValue) {  
312   for (let [key, value] of map.entries()) {  
313     if (value.id === searchValue)  
314       return value.i;  
315   }  
316 }  
317 //---  
318  
319 var init_data = function (){  
320   graph.links.forEach(function(d,i){  
321     d.i = i;  
322   });  
323  
324   //clear up nodeArea  
325   d3.select('#nodeArea').html("");  
326  
327   graph.nodes.forEach(function(d,i){  
328     var nodeArea = d3.select('#nodeArea').append('div').attr("id","node_"+i).attr("class","nodeBt")  
329       .text(d.id)  
330       .on("click", function(d){  
331         let link = d3.select(this).  
332           .append("div").attr("class","linkBt").  
333           .attr("id","link"+i).  
334           .attr("x1", d.x).  
335           .attr("y1", d.y).  
336           .attr("x2", graph.links[i].x2).  
337           .attr("y2", graph.links[i].y2);  
338         link.append("div").attr("class","linkLine").  
339           .attr("x1", graph.links[i].x1).  
340           .attr("y1", graph.links[i].y1).  
341           .attr("x2", graph.links[i].x2).  
342           .attr("y2", graph.links[i].y2);  
343       })  
344     })  
345   })  
346 }  
347  
348 //function to read data from DF  
349 function readDataFromDF() {  
350   return new Promise((resolve, reject) => {  
351     d3.json("df.json", function(error, data) {  
352       if (error) {  
353         reject(error);  
354       } else {  
355         resolve(data);  
356       }  
357     });  
358   })  
359 }  
360  
361 //function to read data from json file  
362 function d3_json(file, callback) {  
363   d3.json(file, function(error, data) {  
364     if (error) {  
365       console.error(error);  
366     } else {  
367       callback(data);  
368     }  
369   });  
370 }  
371  
372 //function to read data from CSV file  
373 function readDataFromCSV(file) {  
374   return new Promise((resolve, reject) => {  
375     const csvParser = require('fast-csv').parse({  
376       headers: true,  
377       skipEmptyLines: true  
378     });  
379     const fs = require('fs');  
380     const stream = fs.createReadStream(file);  
381     const data = [];  
382     stream.on('data', row => {  
383       data.push(row);  
384     });  
385     stream.on('end', () => {  
386       resolve(data);  
387     });  
388   })  
389 }  
390  
391 //function to read data from DB  
392 function readDataFromDB() {  
393   return new Promise((resolve, reject) => {  
394     const MongoClient = require('mongodb').MongoClient;  
395     const url = "mongodb://127.0.0.1:27017";  
396     const dbName = "graphDB";  
397     const client = new MongoClient(url, {useNewUrlParser: true});  
398     client.connect(err => {  
399       if (err) {  
400         reject(err);  
401       } else {  
402         const db = client.db(dbName);  
403         db.collection("graph").find().toArray((err, result) => {  
404           if (err) {  
405             reject(err);  
406           } else {  
407             resolve(result);  
408           }  
409         });  
410       }  
411     });  
412   })  
413 }  
414  
415 //function to read data from API  
416 function readDataFromAPI() {  
417   return new Promise((resolve, reject) => {  
418     const axios = require('axios');  
419     const url = "https://graph-api.firebaseio.com/graph.json";  
420     axios.get(url).then(response => {  
421       resolve(response.data);  
422     }).catch(error => {  
423       reject(error);  
424     });  
425   })  
426 }  
427  
428 //function to read data from local file  
429 function readDataFromFile(file) {  
430   return new Promise((resolve, reject) => {  
431     const fs = require('fs');  
432     const data = fs.readFileSync(file);  
433     resolve(data);  
434   })  
435 }  
436  
437 //function to read data from URL  
438 function readDataFromURL(url) {  
439   return new Promise((resolve, reject) => {  
440     const axios = require('axios');  
441     axios.get(url).then(response => {  
442       resolve(response.data);  
443     }).catch(error => {  
444       reject(error);  
445     });  
446   })  
447 }  
448  
449 //function to read data from DB  
450 function readDataFromDB() {  
451   return new Promise((resolve, reject) => {  
452     const MongoClient = require('mongodb').MongoClient;  
453     const url = "mongodb://127.0.0.1:27017";  
454     const dbName = "graphDB";  
455     const client = new MongoClient(url, {useNewUrlParser: true});  
456     client.connect(err => {  
457       if (err) {  
458         reject(err);  
459       } else {  
460         const db = client.db(dbName);  
461         db.collection("graph").find().toArray((err, result) => {  
462           if (err) {  
463             reject(err);  
464           } else {  
465             resolve(result);  
466           }  
467         });  
468       }  
469     });  
470   })  
471 }  
472  
473 //function to read data from API  
474 function readDataFromAPI() {  
475   return new Promise((resolve, reject) => {  
476     const axios = require('axios');  
477     const url = "https://graph-api.firebaseio.com/graph.json";  
478     axios.get(url).then(response => {  
479       resolve(response.data);  
480     }).catch(error => {  
481       reject(error);  
482     });  
483   })  
484 }  
485  
486 //function to read data from local file  
487 function readDataFromFile(file) {  
488   return new Promise((resolve, reject) => {  
489     const fs = require('fs');  
490     const data = fs.readFileSync(file);  
491     resolve(data);  
492   })  
493 }  
494  
495 //function to read data from URL  
496 function readDataFromURL(url) {  
497   return new Promise((resolve, reject) => {  
498     const axios = require('axios');  
499     axios.get(url).then(response => {  
500       resolve(response.data);  
501     }).catch(error => {  
502       reject(error);  
503     });  
504   })  
505 }  
506  
507 //function to read data from DB  
508 function readDataFromDB() {  
509   return new Promise((resolve, reject) => {  
510     const MongoClient = require('mongodb').MongoClient;  
511     const url = "mongodb://127.0.0.1:27017";  
512     const dbName = "graphDB";  
513     const client = new MongoClient(url, {useNewUrlParser: true});  
514     client.connect(err => {  
515       if (err) {  
516         reject(err);  
517       } else {  
518         const db = client.db(dbName);  
519         db.collection("graph").find().toArray((err, result) => {  
520           if (err) {  
521             reject(err);  
522           } else {  
523             resolve(result);  
524           }  
525         });  
526       }  
527     });  
528   })  
529 }  
530  
531 //function to read data from API  
532 function readDataFromAPI() {  
533   return new Promise((resolve, reject) => {  
534     const axios = require('axios');  
535     const url = "https://graph-api.firebaseio.com/graph.json";  
536     axios.get(url).then(response => {  
537       resolve(response.data);  
538     }).catch(error => {  
539       reject(error);  
540     });  
541   })  
542 }  
543  
544 //function to read data from local file  
545 function readDataFromFile(file) {  
546   return new Promise((resolve, reject) => {  
547     const fs = require('fs');  
548     const data = fs.readFileSync(file);  
549     resolve(data);  
550   })  
551 }  
552  
553 //function to read data from URL  
554 function readDataFromURL(url) {  
555   return new Promise((resolve, reject) => {  
556     const axios = require('axios');  
557     axios.get(url).then(response => {  
558       resolve(response.data);  
559     }).catch(error => {  
560       reject(error);  
561     });  
562   })  
563 }  
564  
565 //function to read data from DB  
566 function readDataFromDB() {  
567   return new Promise((resolve, reject) => {  
568     const MongoClient = require('mongodb').MongoClient;  
569     const url = "mongodb://127.0.0.1:27017";  
570     const dbName = "graphDB";  
571     const client = new MongoClient(url, {useNewUrlParser: true});  
572     client.connect(err => {  
573       if (err) {  
574         reject(err);  
575       } else {  
576         const db = client.db(dbName);  
577         db.collection("graph").find().toArray((err, result) => {  
578           if (err) {  
579             reject(err);  
580           } else {  
581             resolve(result);  
582           }  
583         });  
584       }  
585     });  
586   })  
587 }  
588  
589 //function to read data from API  
590 function readDataFromAPI() {  
591   return new Promise((resolve, reject) => {  
592     const axios = require('axios');  
593     const url = "https://graph-api.firebaseio.com/graph.json";  
594     axios.get(url).then(response => {  
595       resolve(response.data);  
596     }).catch(error => {  
597       reject(error);  
598     });  
599   })  
600 }  
601  
602 //function to read data from local file  
603 function readDataFromFile(file) {  
604   return new Promise((resolve, reject) => {  
605     const fs = require('fs');  
606     const data = fs.readFileSync(file);  
607     resolve(data);  
608   })  
609 }  
610  
611 //function to read data from URL  
612 function readDataFromURL(url) {  
613   return new Promise((resolve, reject) => {  
614     const axios = require('axios');  
615     axios.get(url).then(response => {  
616       resolve(response.data);  
617     }).catch(error => {  
618       reject(error);  
619     });  
620   })  
621 }  
622  
623 //function to read data from DB  
624 function readDataFromDB() {  
625   return new Promise((resolve, reject) => {  
626     const MongoClient = require('mongodb').MongoClient;  
627     const url = "mongodb://127.0.0.1:27017";  
628     const dbName = "graphDB";  
629     const client = new MongoClient(url, {useNewUrlParser: true});  
630     client.connect(err => {  
631       if (err) {  
632         reject(err);  
633       } else {  
634         const db = client.db(dbName);  
635         db.collection("graph").find().toArray((err, result) => {  
636           if (err) {  
637             reject(err);  
638           } else {  
639             resolve(result);  
640           }  
641         });  
642       }  
643     });  
644   })  
645 }  
646  
647 //function to read data from API  
648 function readDataFromAPI() {  
649   return new Promise((resolve, reject) => {  
650     const axios = require('axios');  
651     const url = "https://graph-api.firebaseio.com/graph.json";  
652     axios.get(url).then(response => {  
653       resolve(response.data);  
654     }).catch(error => {  
655       reject(error);  
656     });  
657   })  
658 }  
659  
660 //function to read data from local file  
661 function readDataFromFile(file) {  
662   return new Promise((resolve, reject) => {  
663     const fs = require('fs');  
664     const data = fs.readFileSync(file);  
665     resolve(data);  
666   })  
667 }  
668  
669 //function to read data from URL  
670 function readDataFromURL(url) {  
671   return new Promise((resolve, reject) => {  
672     const axios = require('axios');  
673     axios.get(url).then(response => {  
674       resolve(response.data);  
675     }).catch(error => {  
676       reject(error);  
677     });  
678   })  
679 }  
680  
681 //function to read data from DB  
682 function readDataFromDB() {  
683   return new Promise((resolve, reject) => {  
684     const MongoClient = require('mongodb').MongoClient;  
685     const url = "mongodb://127.0.0.1:27017";  
686     const dbName = "graphDB";  
687     const client = new MongoClient(url, {useNewUrlParser: true});  
688     client.connect(err => {  
689       if (err) {  
690         reject(err);  
691       } else {  
692         const db = client.db(dbName);  
693         db.collection("graph").find().toArray((err, result) => {  
694           if (err) {  
695             reject(err);  
696           } else {  
697             resolve(result);  
698           }  
699         });  
700       }  
701     });  
702   })  
703 }  
704  
705 //function to read data from API  
706 function readDataFromAPI() {  
707   return new Promise((resolve, reject) => {  
708     const axios = require('axios');  
709     const url = "https://graph-api.firebaseio.com/graph.json";  
710     axios.get(url).then(response => {  
711       resolve(response.data);  
712     }).catch(error => {  
713       reject(error);  
714     });  
715   })  
716 }  
717  
718 //function to read data from local file  
719 function readDataFromFile(file) {  
720   return new Promise((resolve, reject) => {  
721     const fs = require('fs');  
722     const data = fs.readFileSync(file);  
723     resolve(data);  
724   })  
725 }  
726  
727 //function to read data from URL  
728 function readDataFromURL(url) {  
729   return new Promise((resolve, reject) => {  
730     const axios = require('axios');  
731     axios.get(url).then(response => {  
732       resolve(response.data);  
733     }).catch(error => {  
734       reject(error);  
735     });  
736   })  
737 }  
738  
739 //function to read data from DB  
740 function readDataFromDB() {  
741   return new Promise((resolve, reject) => {  
742     const MongoClient = require('mongodb').MongoClient;  
743     const url = "mongodb://127.0.0.1:27017";  
744     const dbName = "graphDB";  
745     const client = new MongoClient(url, {useNewUrlParser: true});  
746     client.connect(err => {  
747       if (err) {  
748         reject(err);  
749       } else {  
750         const db = client.db(dbName);  
751         db.collection("graph").find().toArray((err, result) => {  
752           if (err) {  
753             reject(err);  
754           } else {  
755             resolve(result);  
756           }  
757         });  
758       }  
759     });  
760   })  
761 }  
762  
763 //function to read data from API  
764 function readDataFromAPI() {  
765   return new Promise((resolve, reject) => {  
766     const axios = require('axios');  
767     const url = "https://graph-api.firebaseio.com/graph.json";  
768     axios.get(url).then(response => {  
769       resolve(response.data);  
770     }).catch(error => {  
771       reject(error);  
772     });  
773   })  
774 }  
775  
776 //function to read data from local file  
777 function readDataFromFile(file) {  
778   return new Promise((resolve, reject) => {  
779     const fs = require('fs');  
780     const data = fs.readFileSync(file);  
781     resolve(data);  
782   })  
783 }  
784  
785 //function to read data from URL  
786 function readDataFromURL(url) {  
787   return new Promise((resolve, reject) => {  
788     const axios = require('axios');  
789     axios.get(url).then(response => {  
790       resolve(response.data);  
791     }).catch(error => {  
792       reject(error);  
793     });  
794   })  
795 }  
796  
797 //function to read data from DB  
798 function readDataFromDB() {  
799   return new Promise((resolve, reject) => {  
800     const MongoClient = require('mongodb').MongoClient;  
801     const url = "mongodb://127.0.0.1:27017";  
802     const dbName = "graphDB";  
803     const client = new MongoClient(url, {useNewUrlParser: true});  
804     client.connect(err => {  
805       if (err) {  
806         reject(err);  
807       } else {  
808         const db = client.db(dbName);  
809         db.collection("graph").find().toArray((err, result) => {  
810           if (err) {  
811             reject(err);  
812           } else {  
813             resolve(result);  
814           }  
815         });  
816       }  
817     });  
818   })  
819 }  
820  
821 //function to read data from API  
822 function readDataFromAPI() {  
823   return new Promise((resolve, reject) => {  
824     const axios = require('axios');  
825     const url = "https://graph-api.firebaseio.com/graph.json";  
826     axios.get(url).then(response => {  
827       resolve(response.data);  
828     }).catch(error => {  
829       reject(error);  
830     });  
831   })  
832 }  
833  
834 //function to read data from local file  
835 function readDataFromFile(file) {  
836   return new Promise((resolve, reject) => {  
837     const fs = require('fs');  
838     const data = fs.readFileSync(file);  
839     resolve(data);  
840   })  
841 }  
842  
843 //function to read data from URL  
844 function readDataFromURL(url) {  
845   return new Promise((resolve, reject) => {  
846     const axios = require('axios');  
847     axios.get(url).then(response => {  
848       resolve(response.data);  
849     }).catch(error => {  
850       reject(error);  
851     });  
852   })  
853 }  
854  
855 //function to read data from DB  
856 function readDataFromDB() {  
857   return new Promise((resolve, reject) => {  
858     const MongoClient = require('mongodb').MongoClient;  
859     const url = "mongodb://127.0.0.1:27017";  
860     const dbName = "graphDB";  
861     const client = new MongoClient(url, {useNewUrlParser: true});  
862     client.connect(err => {  
863       if (err) {  
864         reject(err);  
865       } else {  
866         const db = client.db(dbName);  
867         db.collection("graph").find().toArray((err, result) => {  
868           if (err) {  
869             reject(err);  
870           } else {  
871             resolve(result);  
872           }  
873         });  
874       }  
875     });  
876   })  
877 }  
878  
879 //function to read data from API  
880 function readDataFromAPI() {  
881   return new Promise((resolve, reject) => {  
882     const axios = require('axios');  
883     const url = "https://graph-api.firebaseio.com/graph.json";  
884     axios.get(url).then(response => {  
885       resolve(response.data);  
886     }).catch(error => {  
887       reject(error);  
888     });  
889   })  
890 }  
891  
892 //function to read data from local file  
893 function readDataFromFile(file) {  
894   return new Promise((resolve, reject) => {  
895     const fs = require('fs');  
896     const data = fs.readFileSync(file);  
897     resolve(data);  
898   })  
899 }  
900  
901 //function to read data from URL  
902 function readDataFromURL(url) {  
903   return new Promise((resolve, reject) => {  
904     const axios = require('axios');  
905     axios.get(url).then(response => {  
906       resolve(response.data);  
907     }).catch(error => {  
908       reject(error);  
909     });  
910   })  
911 }  
912  
913 //function to read data from DB  
914 function readDataFromDB() {  
915   return new Promise((resolve, reject) => {  
916     const MongoClient = require('mongodb').MongoClient;  
917     const url = "mongodb://127.0.0.1:27017";  
918     const dbName = "graphDB";  
919     const client = new MongoClient(url, {useNewUrlParser: true});  
920     client.connect(err => {  
921       if (err) {  
922         reject(err);  
923       } else {  
924         const db = client.db(dbName);  
925         db.collection("graph").find().toArray((err, result) => {  
926           if (err) {  
927             reject(err);  
928           } else {  
929             resolve(result);  
930           }  
931         });  
932       }  
933     });  
934   })  
935 }  
936  
937 //function to read data from API  
938 function readDataFromAPI() {  
939   return new Promise((resolve, reject) => {  
940     const axios = require('axios');  
941     const url = "https://graph-api.firebaseio.com/graph.json";  
942     axios.get(url).then(response => {  
943       resolve(response.data);  
944     }).catch(error => {  
945       reject(error);  
946     });  
947   })  
948 }  
949  
950 //function to read data from local file  
951 function readDataFromFile(file) {  
952   return new Promise((resolve, reject) => {  
953     const fs = require('fs');  
954     const data = fs.readFileSync(file);  
955     resolve(data);  
956   })  
957 }  
958  
959 //function to read data from URL  
960 function readDataFromURL(url) {  
961   return new Promise((resolve, reject) => {  
962     const axios = require('axios');  
963     axios.get(url).then(response => {  
964       resolve(response.data);  
965     }).catch(error => {  
966       reject(error);  
967     });  
968   })  
969 }  
970  
971 //function to read data from DB  
972 function readDataFromDB() {  
973   return new Promise((resolve, reject) => {  
974     const MongoClient = require('mongodb').MongoClient;  
975     const url = "mongodb://127.0.0.1:27017";  
976     const dbName = "graphDB";  
977     const client = new MongoClient(url, {useNewUrlParser: true});  
978     client.connect(err => {  
979       if (err) {  
980         reject(err);  
981       } else {  
982         const db = client.db(dbName);  
983         db.collection("graph").find().toArray((err, result) => {  
984           if (err) {  
985             reject(err);  
986           } else {  
987             resolve(result);  
988           }  
989         });  
990       }  
991     });  
992   })  
993 }  
994  
995 //function to read data from API  
996 function readDataFromAPI() {  
997   return new Promise((resolve, reject) => {  
998     const axios = require('axios');  
999     const url = "https://graph-api.firebaseio.com/graph.json";  
1000     axios.get(url).then(response => {  
1001       resolve(response.data);  
1002     }).catch(error => {  
1003       reject(error);  
1004     });  
1005   })  
1006 }  
1007  
1008 //function to read data from local file  
1009 function readDataFromFile(file) {  
1010   return new Promise((resolve, reject) => {  
1011     const fs = require('fs');  
1012     const data = fs.readFileSync(file);  
1013     resolve(data);  
1014   })  
1015 }  
1016  
1017 //function to read data from URL  
1018 function readDataFromURL(url) {  
1019   return new Promise((resolve, reject) => {  
1020     const axios = require('axios');  
1021     axios.get(url).then(response => {  
1022       resolve(response.data);  
1023     }).catch(error => {  
1024       reject(error);  
1025     });  
1026   })  
1027 }  
1028  
1029 //function to read data from DB  
1030 function readDataFromDB() {  
1031   return new Promise((resolve, reject) => {  
1032     const MongoClient = require('mongodb').MongoClient;  
1033     const url = "mongodb://127.0.0.1:27017";  
1034     const dbName = "graphDB";  
1035     const client = new MongoClient(url, {useNewUrlParser: true});  
1036     client.connect(err => {  
1037       if (err) {  
1038         reject(err);  
1039       } else {  
1040         const db = client.db(dbName);  
1041         db.collection("graph").find().toArray((err, result) => {  
1042           if (err) {  
1043             reject(err);  
1044           } else {  
1045             resolve(result);  
1046           }  
1047         });  
1048       }  
1049     });  
1050   })  
1051 }  
1052  
1053 //function to read data from API  
1054 function readDataFromAPI() {  
1055   return new Promise((resolve, reject) => {  
1056     const axios = require('axios');  
1057     const url = "https://graph-api.firebaseio.com/graph.json";  
1058     axios.get(url).then(response => {  
1059       resolve(response.data);  
1060     }).catch(error => {  
1061       reject(error);  
1062     });  
1063   })  
1064 }  
1065  
1066 //function to read data from local file  
1067 function readDataFromFile(file) {  
1068   return new Promise((resolve, reject) => {  
1069     const fs = require('fs');  
1070     const data = fs.readFileSync(file);  
1071     resolve(data);  
1072   })  
1073 }  
1074  
1075 //function to read data from URL  
1076 function readDataFromURL(url) {  
1077   return new Promise((resolve, reject) => {  
1078     const axios = require('axios');  
1079     axios.get(url).then(response => {  
1080       resolve(response.data);  
1081     }).catch(error => {  
1082       reject(error);  
1083     });  
1084   })  
1085 }  
1086  
1087 //function to read data from DB  
1088 function readDataFromDB() {  
1089   return new Promise((resolve, reject) => {  
1090     const MongoClient = require('mongodb').MongoClient;  
1091     const url = "mongodb://127.0.0.1:27017";  
1092     const dbName = "graphDB";  
1093     const client = new MongoClient(url, {useNewUrlParser: true});  
1094     client.connect(err => {  
1095       if (err) {  
1096         reject(err);  
1097       } else {  
1098         const db = client.db(dbName);  
1099         db.collection("graph").find().toArray((err, result) => {  
1100           if (err) {  
1101             reject(err);  
1102           } else {  
1103             resolve(result);  
1104           }  
1105         });  
1106       }  
1107     });  
1108   })  
1109 }  
1110  
1111 //function to read data from API  
1112 function readDataFromAPI() {  
1113   return new Promise((resolve, reject) => {  
1114     const axios = require('axios');  
1115     const url = "https://graph-api.firebaseio.com/graph.json";  
1116     axios.get(url).then(response => {  
1117       resolve(response.data);  
1118     }).catch(error => {  
1119       reject(error);  
1120     });  
1121   })<
```

DATA FOUNDRY

My projects

Data tools

Documentation

Show entries

Support

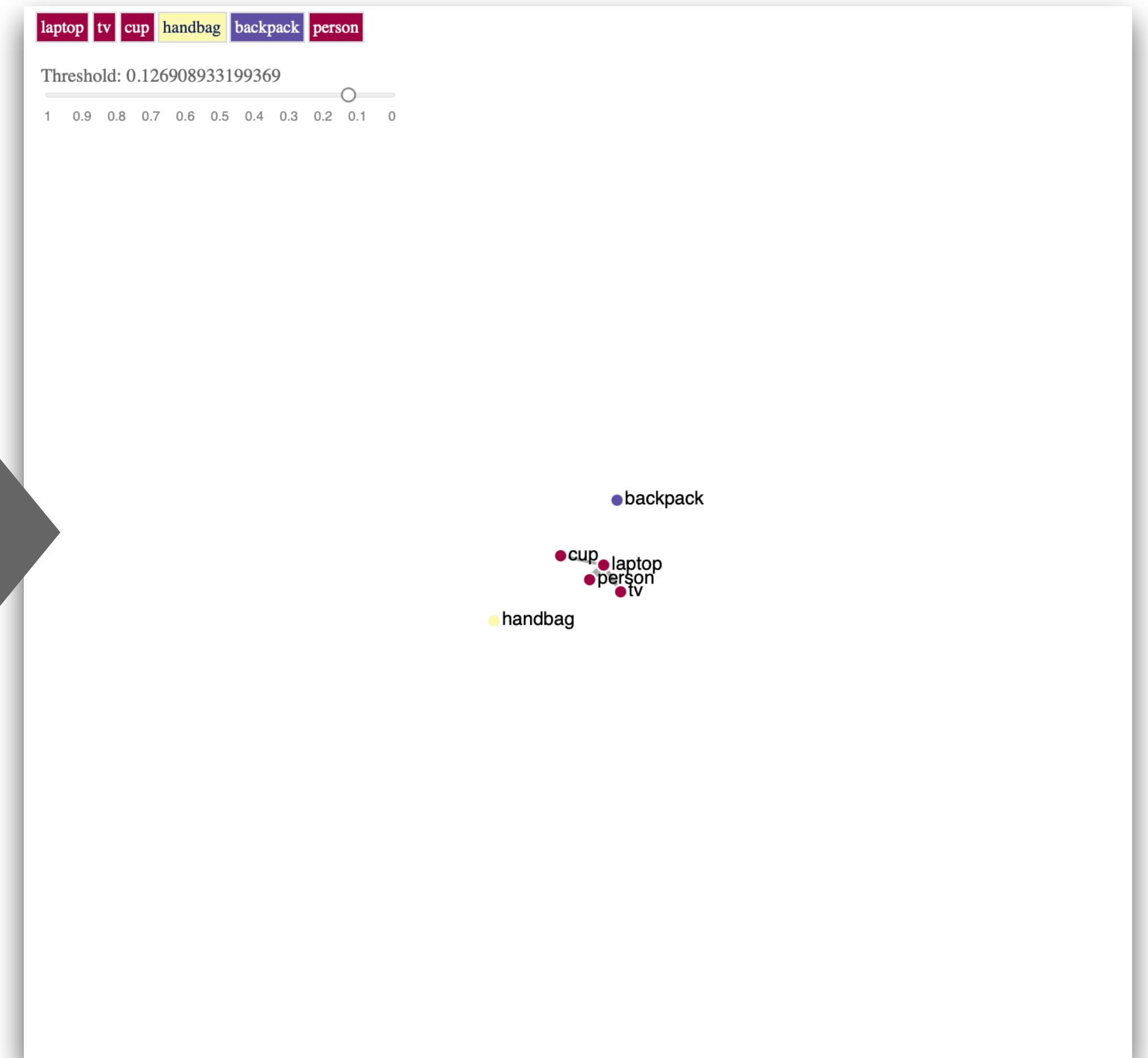
Profile

Logout

Home > [ARTIFICE] AI Workshop 2023: Customize ThingCV > Object data > Data table

OBJECT DATA

	id	ts	pp1	pp2	pp3	detections	time
1	2023-10-02T16:56:03					laptop, tv, tv	1696258563822
2	2023-10-02T16:56:04					laptop, tv, cup, tv	1696258564523
3	2023-10-02T16:56:04					laptop, tv, tv	1696258564827
4	2023-10-02T16:56:05					laptop, tv, tv	1696258565008
5	2023-10-02T16:56:05					laptop, tv, tv	1696258565191
6	2023-10-02T16:56:05					laptop, tv, tv	1696258565374
7	2023-10-02T16:56:05					laptop, tv, tv	1696258565517
8	2023-10-02T16:56:10					tv	1696258570971
9	2023-10-02T16:56:11					tv	1696258571168
10	2023-10-02T16:56:11					tv	1696258571399
11	2023-10-02T16:56:11					tv	1696258571571



Exercise 4: Data Visualizer

Step 1: Create an existing dataset

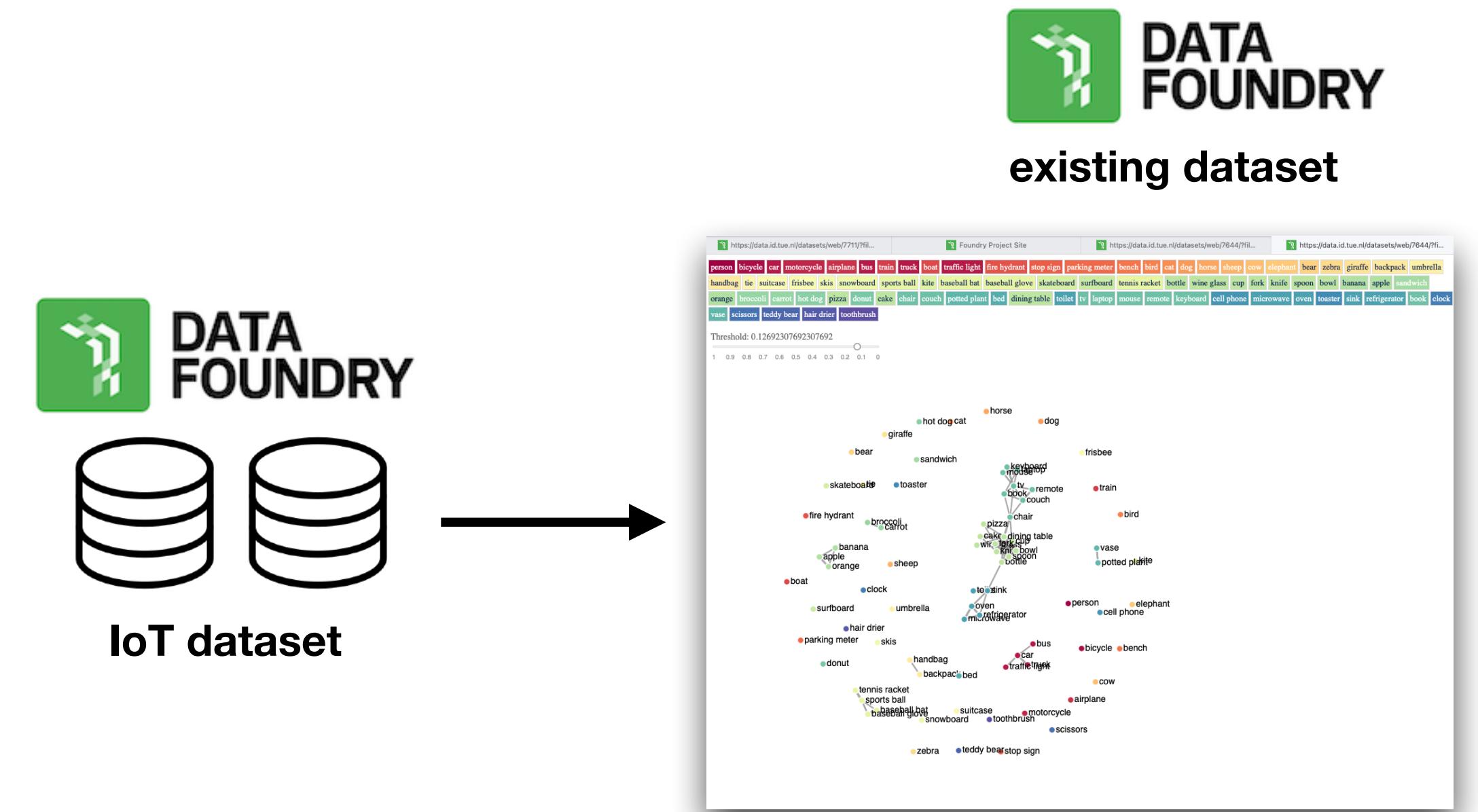
Step 2: Download the source code and upload the code to the existing dataset

Step 3: Check whether the default data visualization is live through the website link

Step 4: Find and copy the data link (JSON link) under the object dataset

Step 5: change the code in “index.html” under data visualizer (in the existing dataset) and change input method (read input data from a json file to the JSON link)

Step 6: Check whether your data visualization that display the actual data stored from the IoT dataset (i.e., object data)



Use DF to host your website that includes a **data visualization**

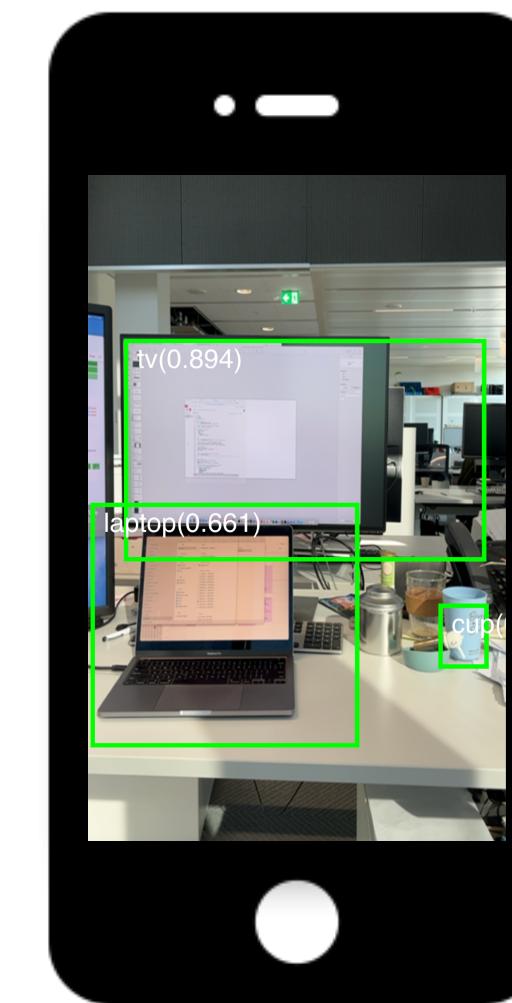
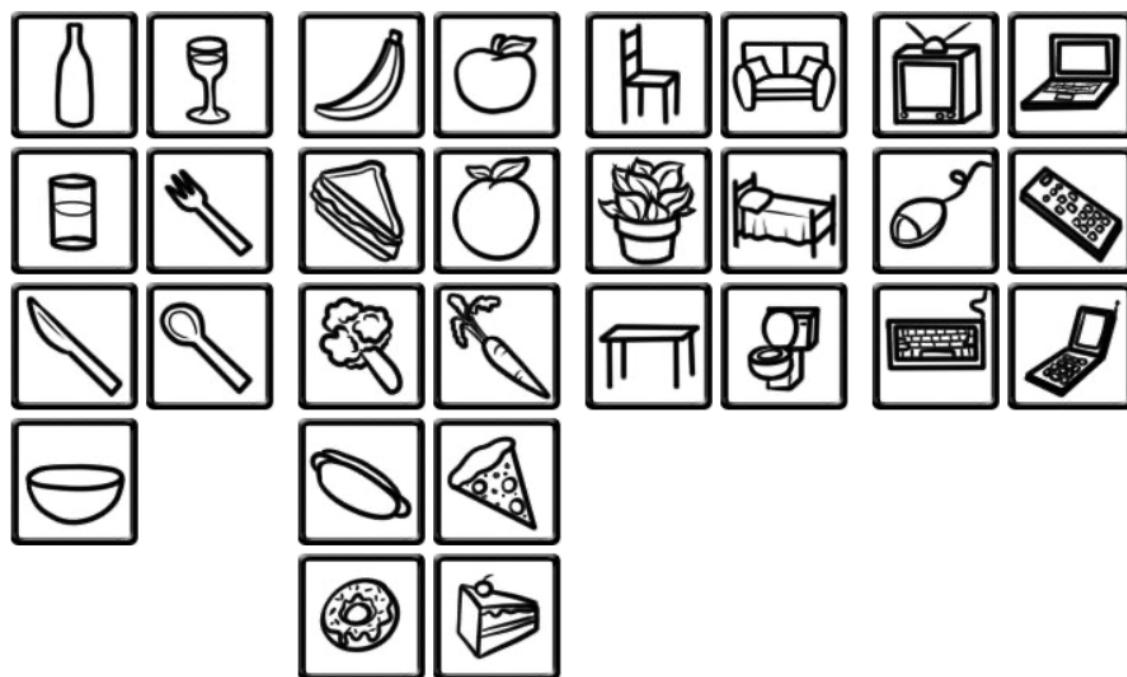


1

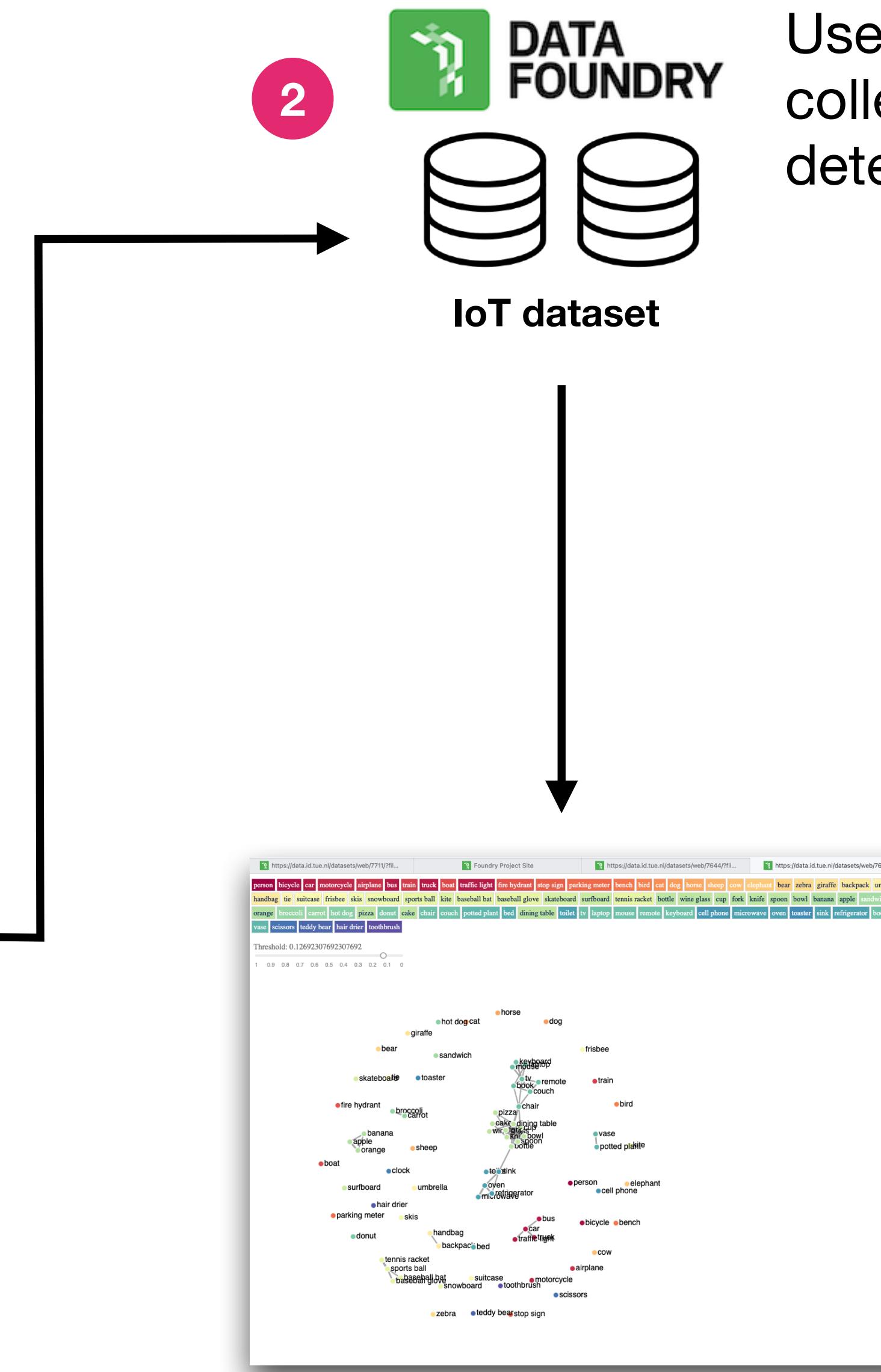
p5.js

ml5

Use DF to host your website
that includes **an object
detector using a pre-trained
model**



Object Detector



Use DF to **store object data**
collected from your object
detector

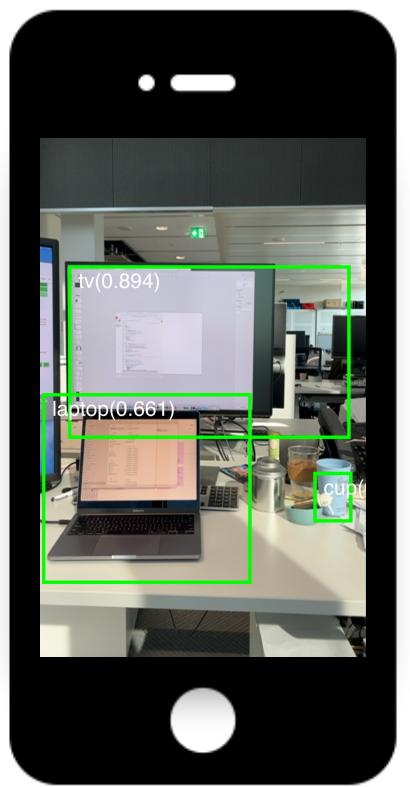


3

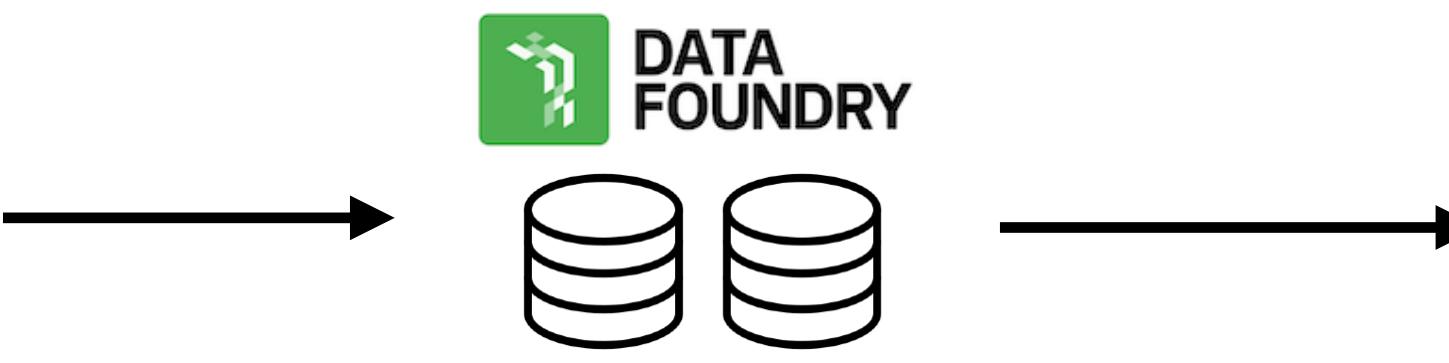
existing dataset

Use DF to host your
website that includes a
data visualization

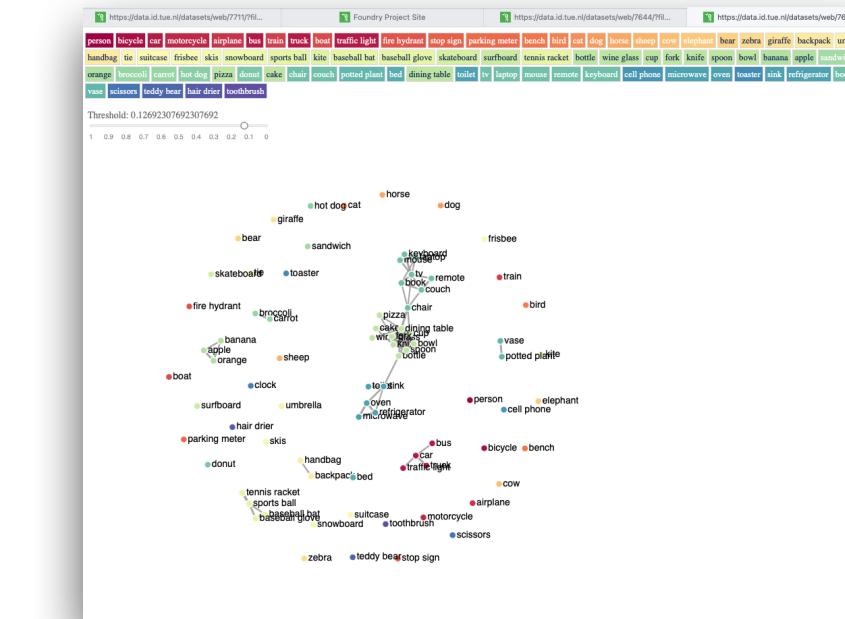
Play with your own thingCV and celebrate! :)



Object Detector



IoT dataset

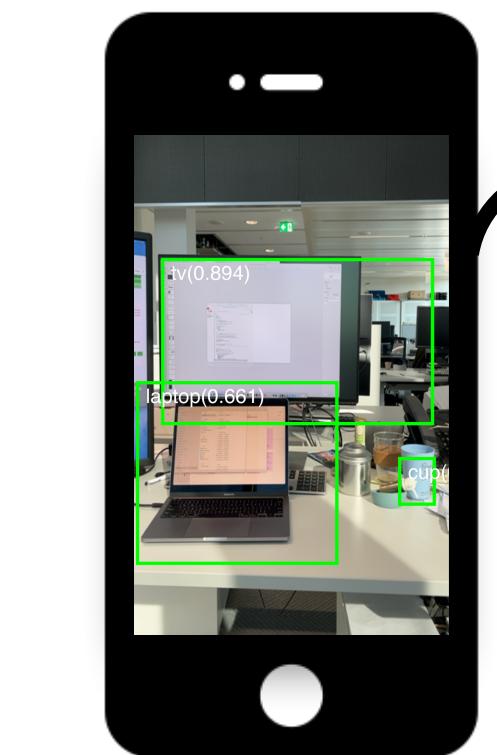
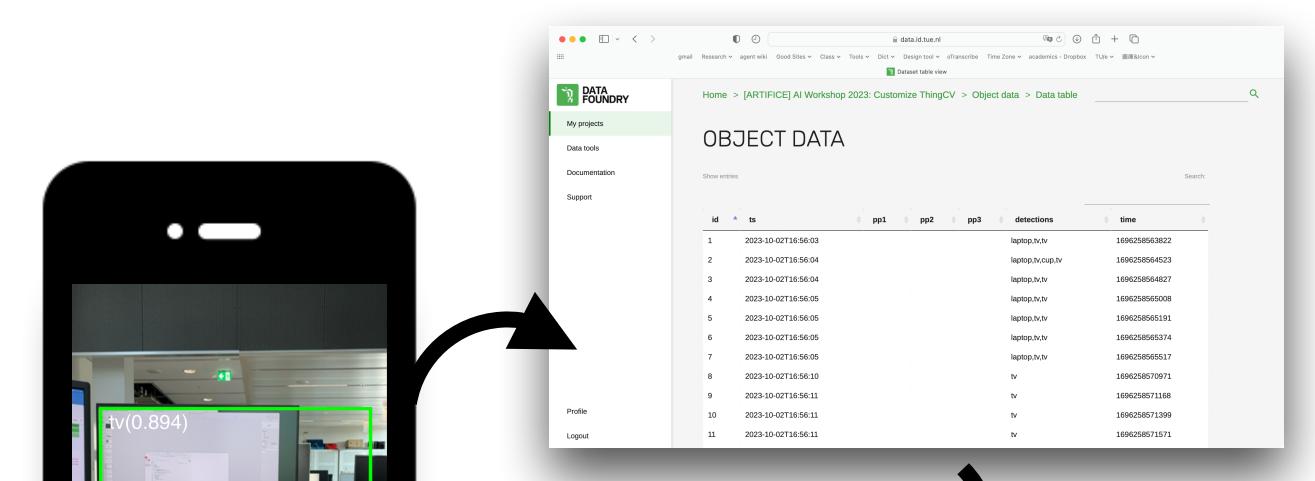
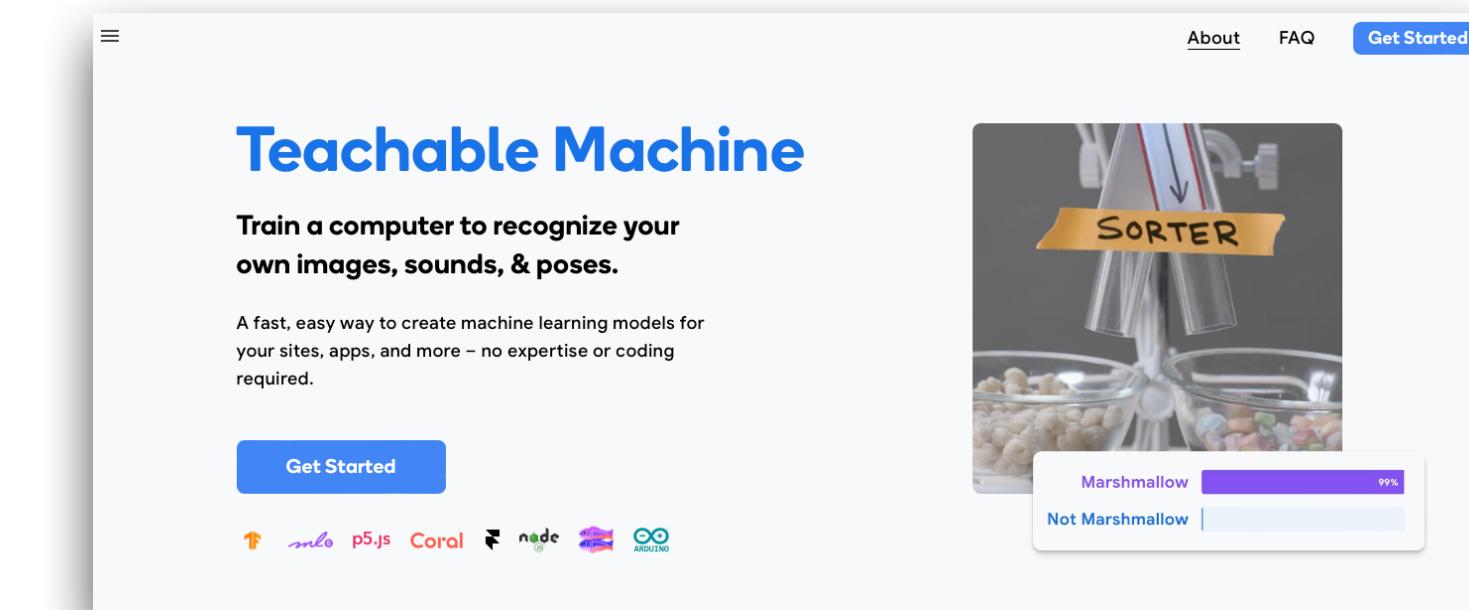


Data Visualizer

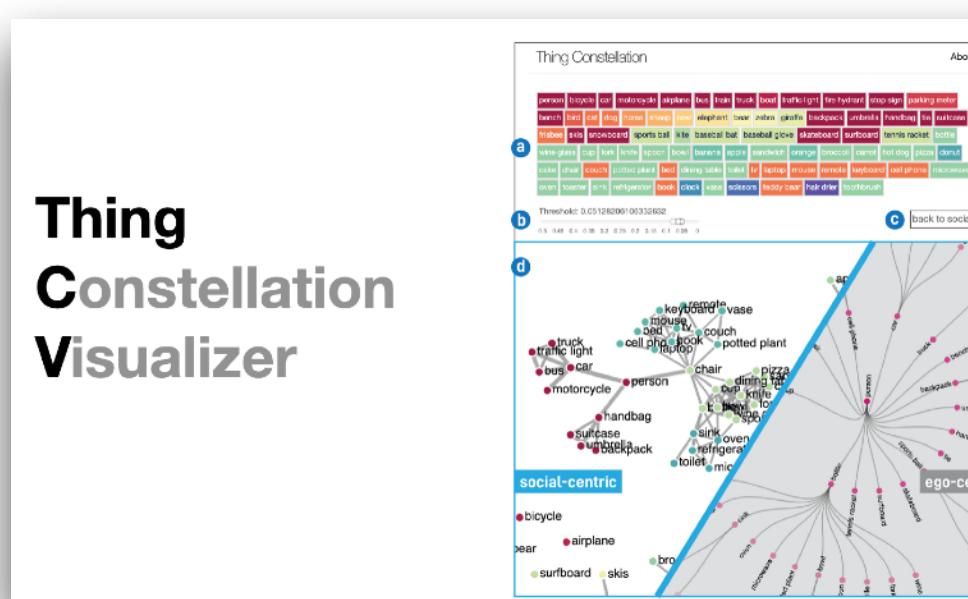
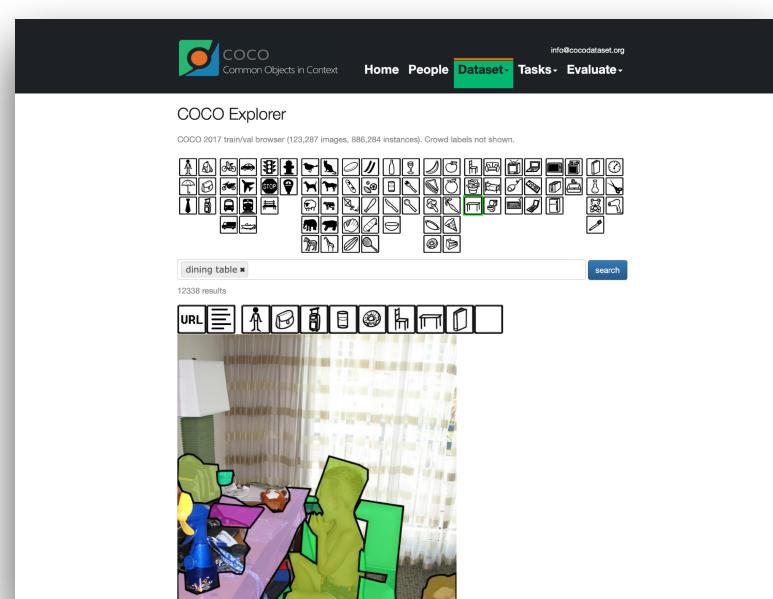
Recap

- use Teachable Machine to **train your own classifier**
- use MS-COCO dataset and thingCV to **support your design exploration**
- use AI canvas to **design the whole system**

- use Data Foundry, p5.js and ml5.js to **build an entire system**
 - including object detector, data storage, and data visualization
- object detector
 - **build an object detector** using a pre-trained model (i.e., COCO-SD or YOLO)
 - host your object detector in DF
- data storage
 - use DF to **store your data** (i.e., AI-extracted data)
- data visualization
 - use DF to **host your interactive data visualization**



Object Detector



Thing
Constellation
Visualizer

A screenshot of the AI Canvas interface. It consists of a 2x4 grid of numbered steps. Step 1: Concept / Idea Description. Step 2: Contextual situations. Step 3: The Role of Human. Step 4: The Role of AI. Step 5: Feature. Step 6: Input Data. Step 7: Output Results. Step 8: Open questions / problems.

Q&A

Thank you!