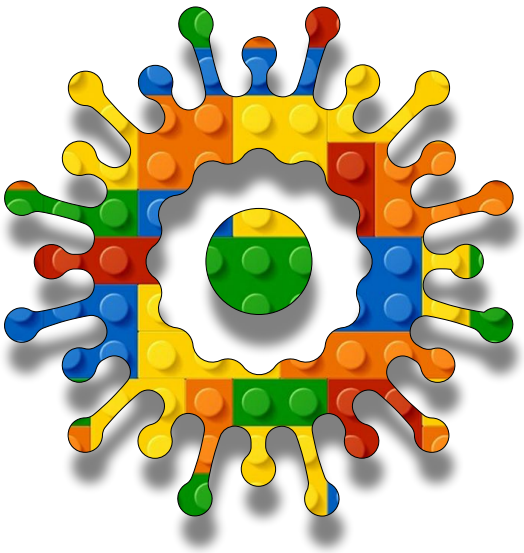
Algorithmic Intelligence Module B Unit #8 mobileNet Video Classification



Content

Module B Unit #8 mobileNet video classification

Introduction to mobileNet video classification

Using `mobileNet` as before we are going to live stream and try to classify what it can see, should be fun! There are a lot of similarities to classifying an image.

Sketch B8.1 index.html

Make sure you have the `ml5.js` line of code in your `index.html` file.

```
<!DOCTYPE html>
<html lang="en"><head>
    <script src="https://cdn.jsdelivr.net/npm/p5@2.2.2/lib/p5.js"></script>
    <script src="https://unpkg.com/ml5@1/dist/ml5.min.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8">
  </head>
  <body>
    <main>
    </main>
    <script src="sketch.js"></script>
</body></html>
```

Sketch B8.2 starting sketch

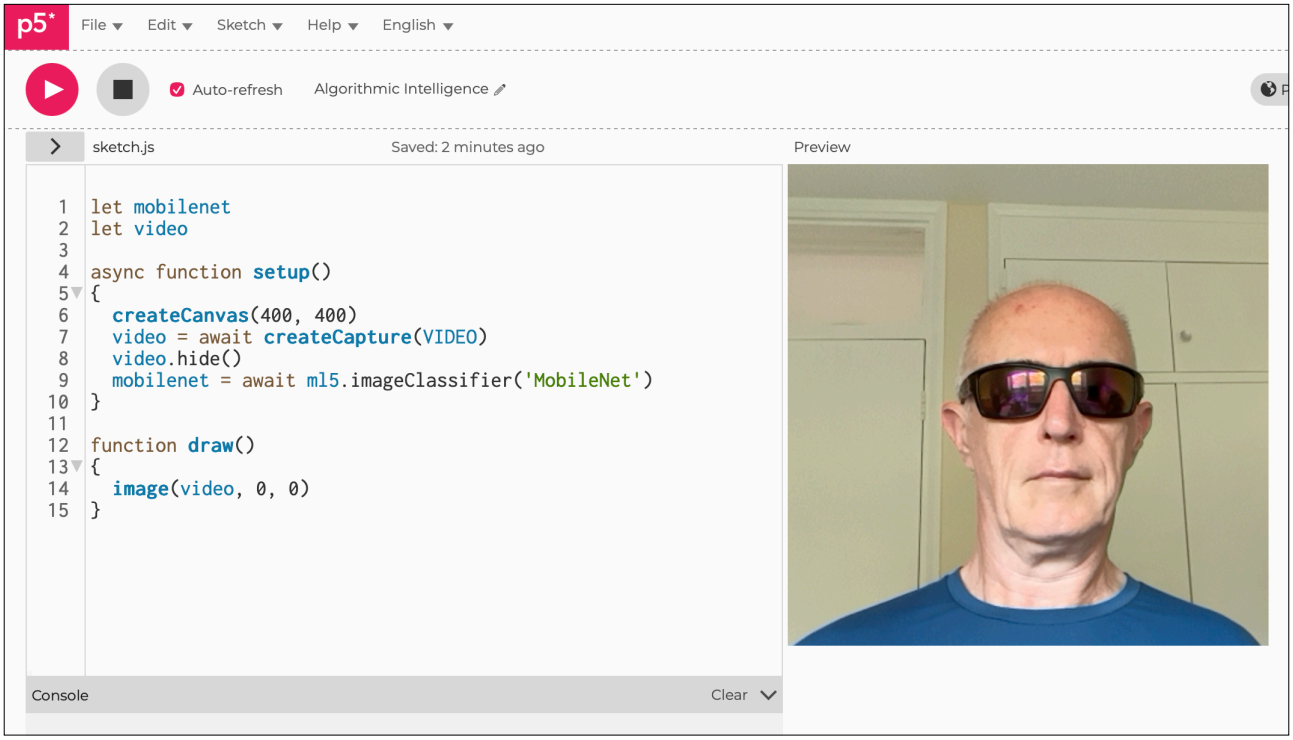
Getting the video to work, you will get a prompt asking permission to allow access to the webcam.

```
let mobilenet
let video

async function setup()
{
  createCanvas(400, 400)
  video = await createCapture(VIDEO)
  video.hide()
  mobilenet = await ml5.imageClassifier('MobileNet')
}

function draw()
{
  image(video, 0, 0)
}
```

Notes

You should get an image of yourself like below, but maybe without the sunglasses. The reason I am wearing sunglasses will be explained shortly.

Figure B8.2


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Intelligence
sketch.js
Saved: 2 minutes ago
Preview
let mobilenet
let video
async function setup()
{
createCanvas(400, 400)
video = await createCapture(VIDEO)
video.hide()
mobilenet = await ml5.imageClassifier('MobileNet')
}
function draw()
{
image(video, 0, 0)
}
Console
Clear

Sketch B8.3 classifying

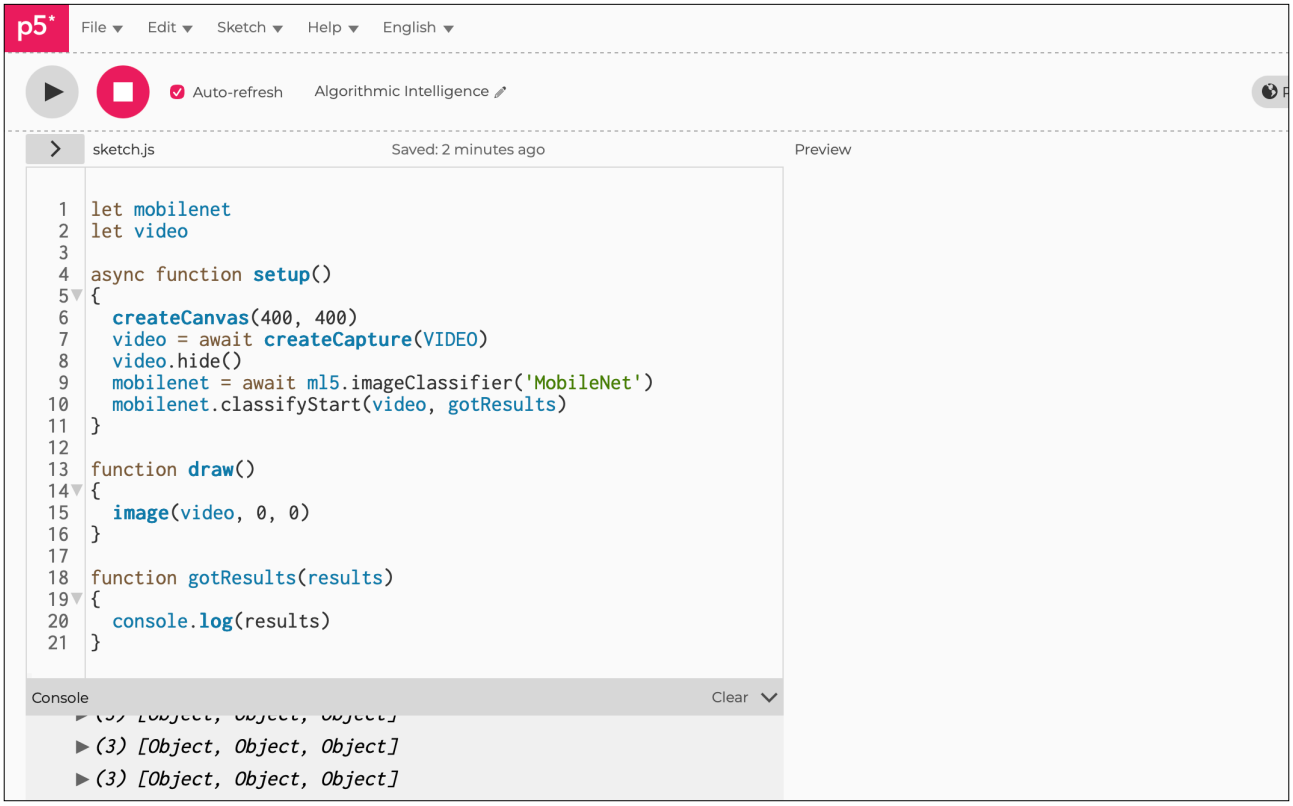
To classify a streaming video, we need to use a different function called `classifyStart()` rather than just `classify()`. We also add our old friend, the `gotResults()` function.

```
let mobilenet
let video

async function setup()
{
  createCanvas(400, 400)
  video = await createCapture(VIDEO)
  video.hide()
  mobilenet = await ml5.imageClassifier('MobileNet')
  mobilenet.classifyStart(video, gotResults)
}

function draw()
{
  image(video, 0, 0)
}

function gotResults(results)
{
  console.log(results)
}
```

Notes

We get a continuous stream of array data, so press the stop/pause button and take a look inside one of these arrays. You will see three predictions with the usual label and confidence score. You might be very surprised by what it thinks it is looking at!

Figure B8.3


p5*
File
Edit
Sketch
Help
English
Auto-refresh
Algorithmic Intelligence
sketch.js
Saved: 2 minutes ago
Preview
let mobilenet
let video
async function setup()
{
createCanvas(400, 400)
video = await createCapture(VIDEO)
video.hide()
mobilenet = await ml5.imageClassifier('MobileNet')
mobilenet.classifyStart(video, gotResults)
}
function draw()
{
image(video, 0, 0)
}
function gotResults(results)
{
console.log(results)
}
Console
Clear
▶ (3) [Object, Object, Object]
▶ (3) [Object, Object, Object]

Sketch B8.4 display the results

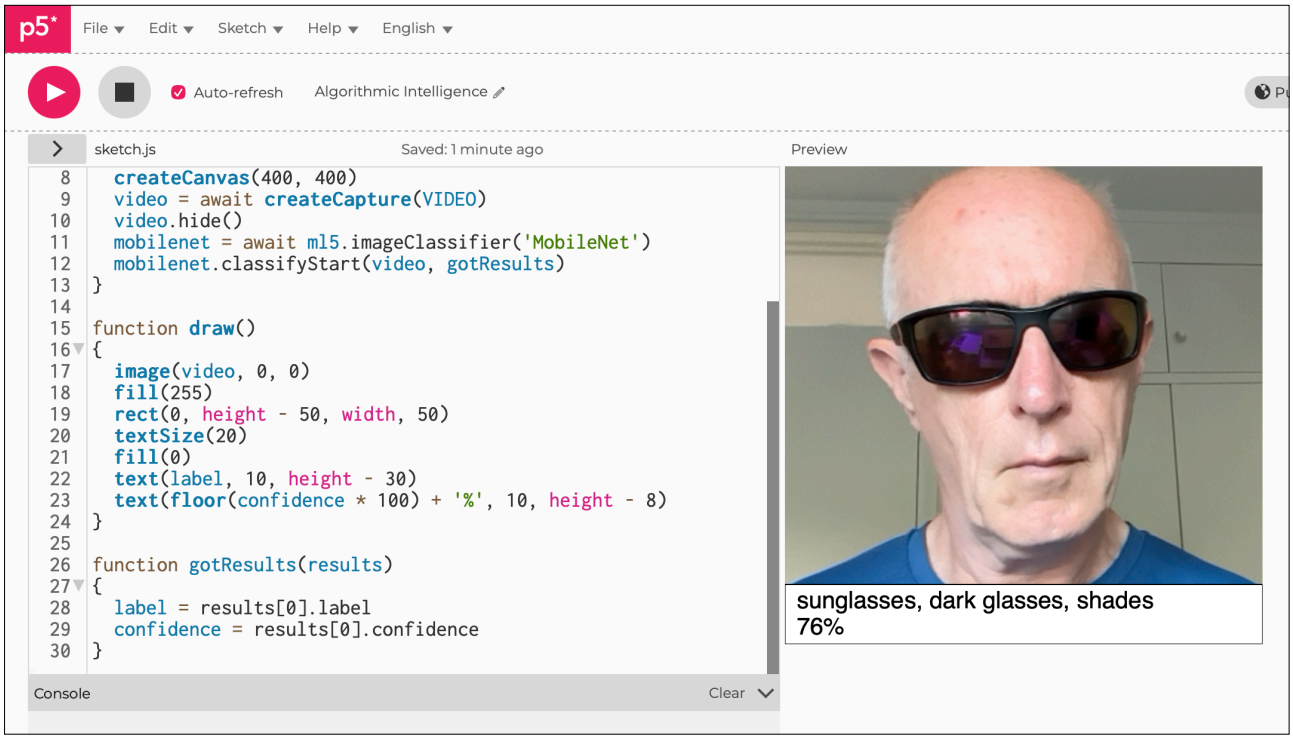
We can follow a similar trend as we did with the image, but to make sure we can see the results, I will put the text over a separate background.

```
let mobilenet
let video
let label = ''
let confidence = ''

async function setup()
{
  createCanvas(400, 400)
  video = await createCapture(VIDEO)
  video.hide()
  mobilenet = await ml5.imageClassifier('MobileNet')
  mobilenet.classifyStart(video, gotResults)
}

function draw()
{
  image(video, 0, 0)
  fill(255)
  rect(0, height - 50, width, 50)
  textSize(20)
  fill(0)
  text(label, 10, height - 30)
  text(floor(confidence * 100) + '%', 10, height - 8)
}

function gotResults(results)
{
  label = results[0].label
  confidence = results[0].confidence
}
```

Notes

This is why I wore the sunglasses; it gave a very correct answer, even if the confidence value varied from 30% to 80% constantly. Some of the other predictions were a lot less flattering, to say the least.

Figure B8.4



Sketch B8.5 a bit of a mug

By adding a button and changing the predicting function from `classifyStart()` to `classify()` and then putting it in another function called `snapshot()`, we can now position something and classify it just once.

```
let mobilenet
let video
let label = ''
let confidence = ''
let button

async function setup()
{
  createCanvas(400, 400)
  video = await createCapture(VIDEO)
  video.hide()
  mobilenet = await ml5.imageClassifier('MobileNet')
  // mobilenet.classifyStart(video, gotResults)
  button = createButton('take a snapshot')
  button.mousePressed(snapshot)
}

function snapshot()
{
  mobilenet.classify(video, gotResults)
}

function draw()
{
  image(video, 0, 0)
  fill(255)
  rect(0, height - 50, width, 50)
  textSize(20)
  fill(0)
  text(label, 10, height - 30)
  text(floor(confidence * 100) + '%', 10, height - 8)
}
```

```
function gotResults(results)
{
  label = results[0].label
  confidence = results[0].confidence
}
```

Notes

This is repeatable, so get another object and click on the button.

Figure B8.5