

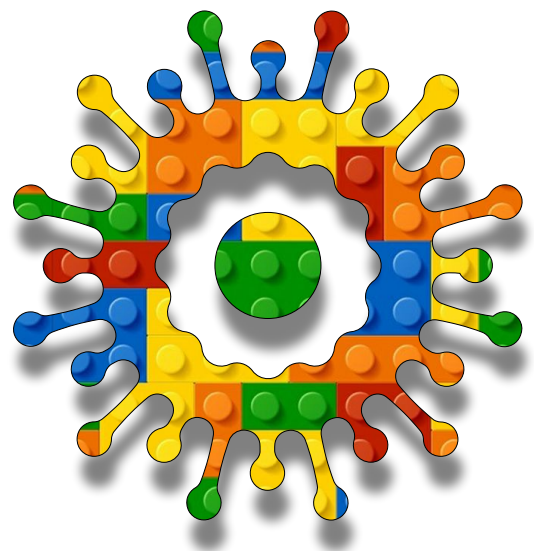
# Algorithmic Intelligence

## Module C

### Unit #3

## transformer

## depth analysis





## Module C Unit #3 transformer depth analysis

Sketch C3.1 the code



# Introduction to transformer depth analysis

There are numerous tasks. Another one is depth analysis. We follow the same procedure as before. We will use the same photo as before, so keep most of the code from before. Here, it will try to estimate depth in the photo. The brighter the colour, the nearer it is.

I changed the task to depth analysis and scrolled down till I found one I liked the look of. In this case, I chose:

[onnx-community/depth-anything-v2-small](#)

I recommend that you look it up yourself and see how I got the code to work.

Figure 1: depth analysis

The screenshot shows the Hugging Face model page for `onnx-community/depth-anything-v2-small`. The page includes a search bar, navigation links (Models, Datasets, Spaces, Buckets, Docs, Pricing), and a header with the model name, likes (23), and followers (1.77k). The main content area is divided into sections: **Model card**, **Files and versions**, and **Community**. The **Model card** section contains a link to the model page, a description of the model's usage, and a code snippet for installing the `transformers` library and performing depth estimation. The **Files and versions** section shows the model's base model and quantized versions. The **Community** section lists spaces that use the model.

```
npm i @huggingface/transformers
```

```
import { pipeline } from '@huggingface/transformers';

// Create depth estimation pipeline
const depth_estimator = await pipeline('depth-estimation', 'onnx-comm
```

If you haven't still go it in your sketch from the last unit, download the dog image as we have done before, then upload it as a file. We just call it dog (remember to rename it in the file).

Figure 2: download dog image



Figure 3: our canine friend

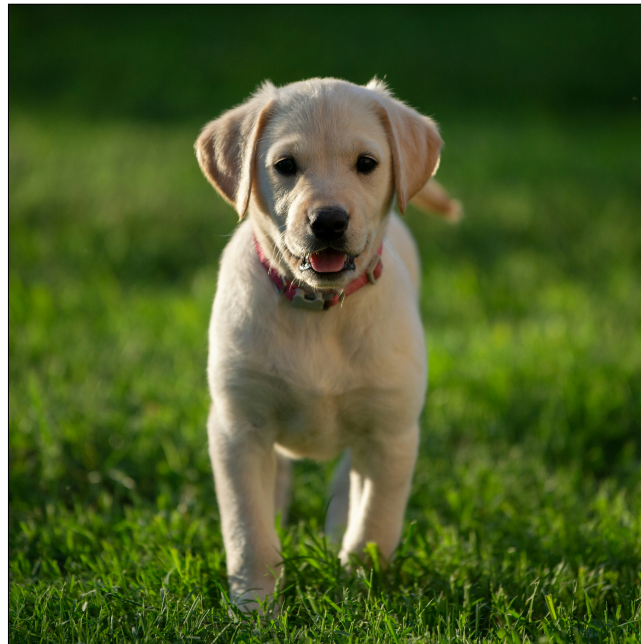
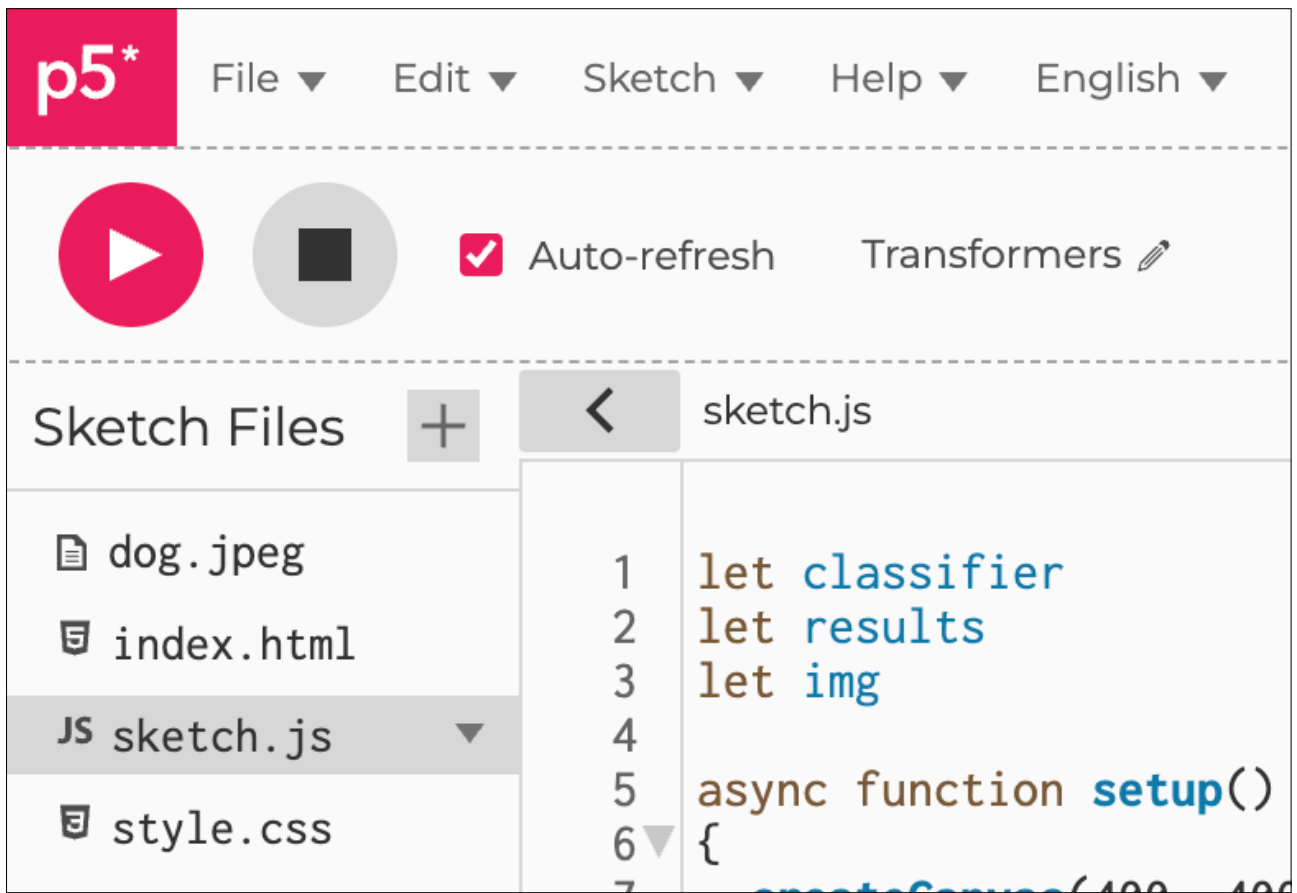


Figure 4: Sketch File (dog added)





## Sketch C3.1 the code

Our starting code is below. Very similar to the last one with a different model. We will save the resulting image as a PNG.

```
let depth_estimator
let img

async function setup()
{
  createCanvas(400, 400)
  background(220)
  img = await loadImage('dog.jpeg')
  image(img, 0, 0, 400, 400)
  let { pipeline } = await import('https://cdn.jsdelivr.net/npm/@huggingface/transformers')
  depth_estimator = await pipeline('depth-estimation', 'onnx-community/depth-anything-v2-small')
  const { depth } = await depth_estimator('dog.jpeg')
  depth.save('depth.png')
}
```



### Notes

The main addition is the `depth.save()` function; this should result in a PNG being uploaded. The image is below:

Figure C3.1

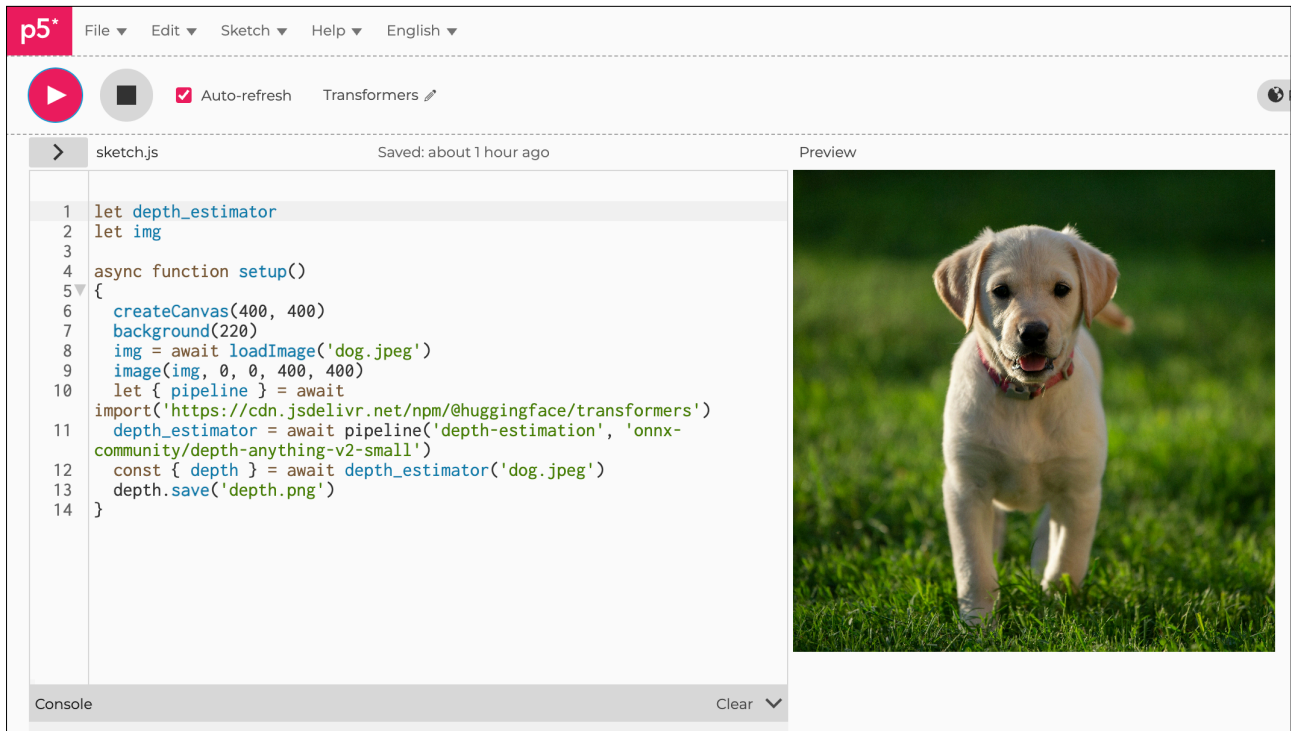


Figure 5: depth analysis

