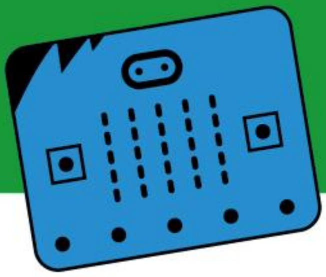
Sheet 1



Sheet 1

1 Presentation

This file includes basic definitions on Artificial Intelligence and Machine Learning and their differences.

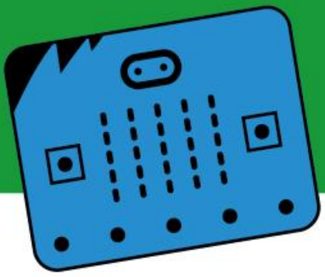
Traditional programming and Machine Learning concepts are presented and contrasted. To apply these concepts, we propose a step-by-step guide on how to create a model using the Teachable Machine platform, which supports data in image and audio format, to create your own projects. Best practice tips are offered to be used in your first steps.

Suggestions are offered to continue practicing, and students are encouraged to continue learning in our next sheet.

2 Introduction to Artificial Intelligence and Machine Learning

What are **Artificial Intelligence** and **Machine Learning (ML)**? How do these concepts differ?

In this sheet, we will approach two concepts that usually seem extremely complex: Artificial Intelligence (AI) and Machine Learning (ML). Although it may seem difficult to



create projects with AI or ML, we invite you to get to work; we will accompany you step by step through different tools that will allow you to explore some of the infinite possibilities offered by this universe.

So... what do we talk about when we talk about AI and ML?

AI is a set of skills that a computer has in order to perform activities that simulate human intelligence, that is, the ability to run algorithms, learn from data and use what they learn in decision making just as a person would.

On the other hand, Machine Learning is a subfield of AI, one of its possible applications. More specifically, it is the ability of a computer to learn or recognize patterns. The basis of this skill lies in the entry of data into the system, which will allow it to make predictions and suggestions.

To better understand these concepts, we invite you to get to work.

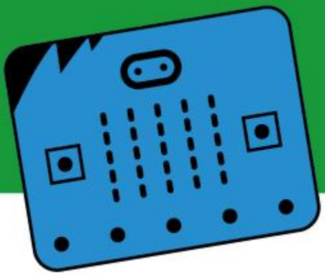
3 Get to work! Traditional Programming versus Machine Learning

The first step is to understand how Traditional Programming and Machine Learning work, and how they are different:

- **Traditional programming:** Data is entered into the system< the system applies a series of rules; an output is obtained. For example, in a calculator you enter numbers and a rule (to add, multiply, divide, subtract, etc.). New data is the output (a new number, the result). In this type of programming, for the same input data, the output will always be the same.

- **Machine Learning:** This type of programming works differently from traditional programming. A certain amount of input and output data is entered into the system. With that information, the system creates the rules. More specifically, a certain amount of input data is assigned and classified in such a way that the computer can identify which category different data (rule) belong to.

Sheet 1



Traditional programming = Input + Rules ⟶ Output

Machine Learning = Inputs + Outputs ⟶ Rules

In traditional programming computers make no mistakes, but when AI is used they may; it all depends on how the computer is trained or, to be more precise, how the **model** is trained.

The result of the “training” of a computer or model can be called **Machine Learning**. The computer is given, through algorithms or instructions, the ability to identify “patterns” and, based on these, predict how a given input data should be classified.

3.1 What types of data can be used?

The model may be of various types of data, depending on the environment.
There are three main types:

- images/videos,
- texts,
- audios.

3.2 How to create a Machine Learning model?

One of the tools available for this type of projects and one used in this file is **Teachable Machine**.

On this platform users can generate Machine Learning models according to the project to be carried out. In order to use it, you must access the Teachable Machine website.

The Machine Learning type used by this platform is supervised learning.

This programming environment will allow you to train a model through the following stages:

a. **Data collection** Enter information into the system and classify it by classes or categories.

b. **Model preparation:** Generate and test data operation.

c. **Model export:** Generate a link using the model and then use it with a micro:bit board.

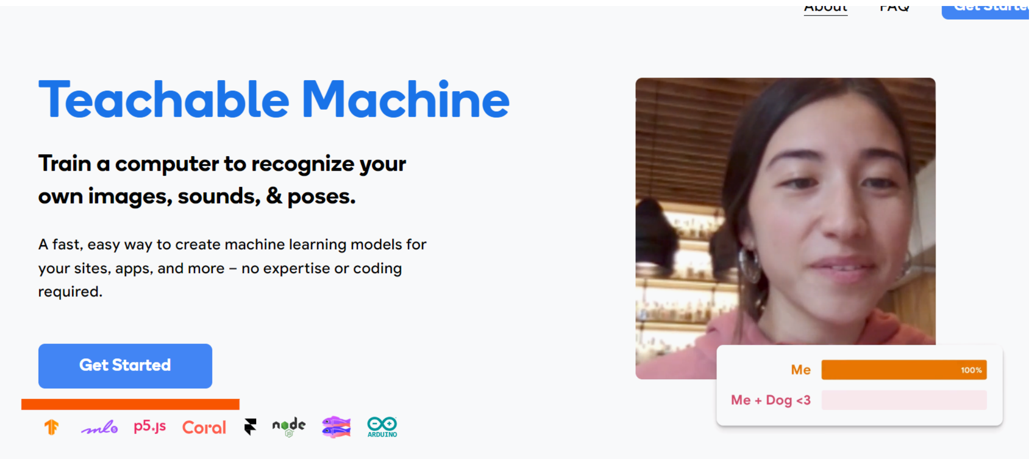
4 Model generation

On the Teachable Machine platform, data tagging refers to assigning data a **category** or **class**. For example, two sets of pictures of dogs and cats are provided. Each of these series or classes is tagged with a title or a name. By doing this, these categories will help the model to establish what are the common characteristics among the data entered for each class, so that it can decide, once a new image is presented, which class it belongs to.

Follow these steps to start your project:

Step 1: Click on **Get Started**

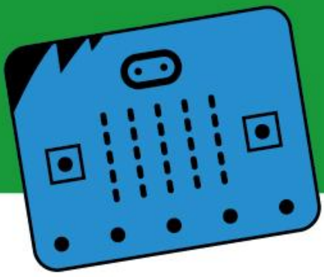
.



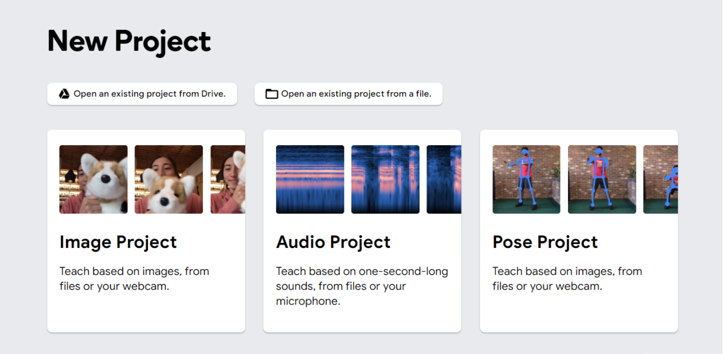
Teachable Machine

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.



Step 2: Choose the type of model to be created.



New Project

Open an existing project from Drive.
Open an existing project from a file.

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.

This environment allows projects to be carried out using three types of data:

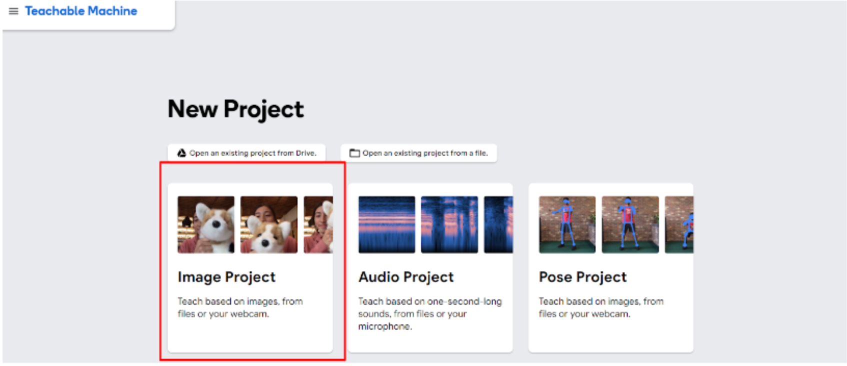
- image
- audio
- poses (through the webcam)

How to build each of these models is explained below. Join us!

4.1 Image Project:

Stage 1:

a) Select "Image Project".



New Project

Open an existing project from Drive.
Open an existing project from a file.

Image Project

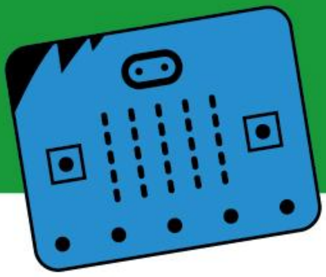
Teach based on images, from files or your webcam.

Audio Project

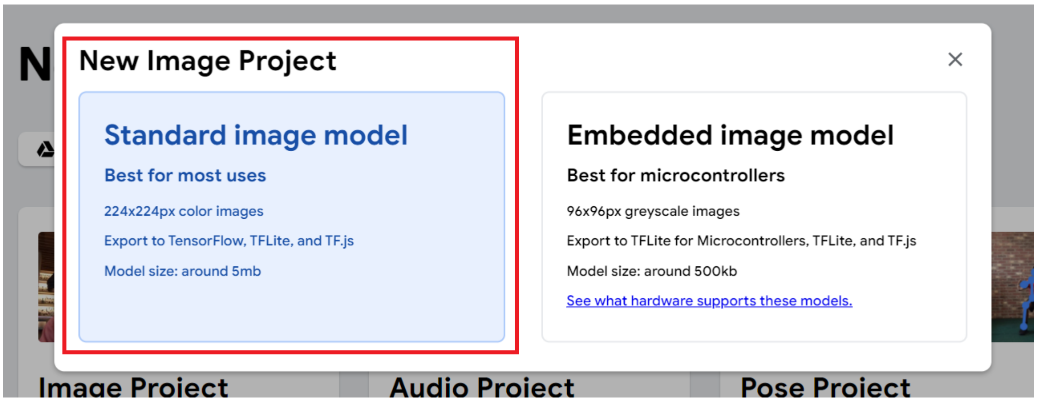
Teach based on one-second-long sounds, from files or your microphone.

Pose Project

Teach based on images, from files or your webcam.



b) Select "Standard Image Model".



N

New Image Project

×

Standard image model

Best for most uses

224x224px color images

Export to TensorFlow, TFLite, and TF.js

Model size: around 5mb

Embedded image model

Best for microcontrollers

96x96px greyscale images

Export to TFLite for Microcontrollers, TFLite, and TF.js

Model size: around 500kb

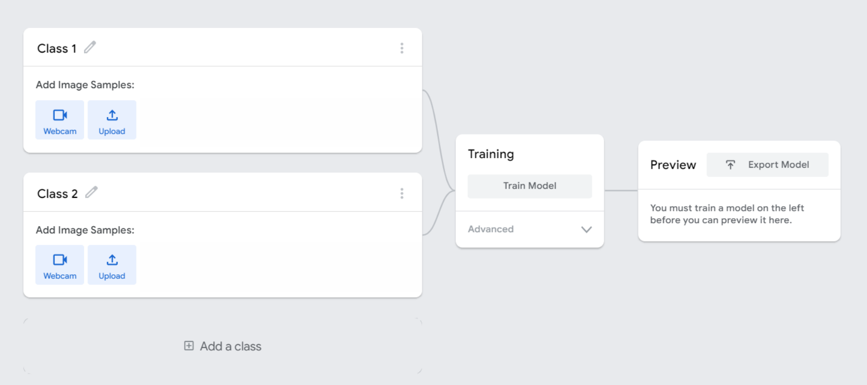
See what hardware supports these models.

Image Project

Audio Project

Pose Project

c) Enter and sort data.



Class 1

Add Image Samples:

Webcam

Upload

Class 2

Add Image Samples:

Webcam

Upload

Add a class

Training

Train Model

Advanced

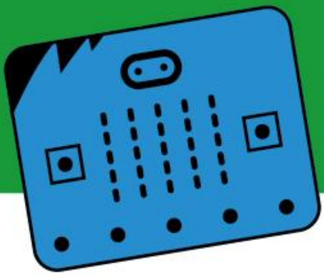
Preview

Export Model

You must train a model on the left before you can preview it here.

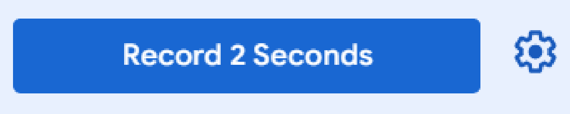
As an example, we can choose categories such as "Right Hand", and place a set of images that show one or more people raising their right hand. Then, in the next category, "Left Hand", we can do the same but with the left hand. Finally, a third category, "Hands Down", will contain images of a person with both hands down.

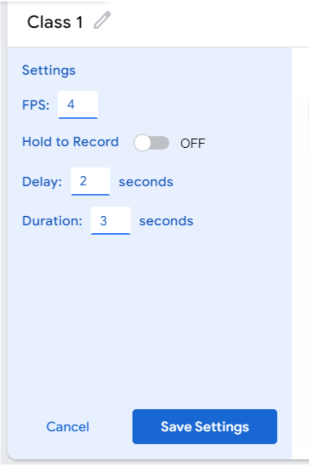
Sheet 1



The application allows you to add images from a camera or from the image database of your device (computer, cell phone, etc.).

If the webcam is used, data collection can be defined through set up. Tip: Take 4 FPS (frames per second) for 3 seconds (12 pieces of data in total). There is a time frame before recording starts; it is advisable to select 2 seconds. Those images that we do not consider representative of the class should be discarded.





Class 1

Settings

FPS: 4

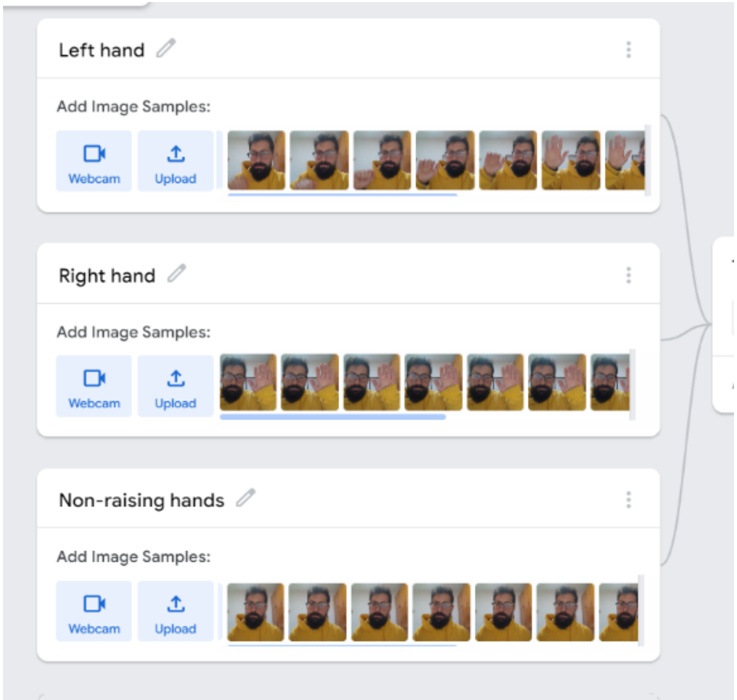
Hold to Record OFF

Delay: 2 seconds

Duration: 3 seconds

In this example, the process is repeated in the three predefined classes. The number of classes to be used will depend on the type of project to be carried out. It is advisable to start with a small number of classes.

Once data have been uploaded to the Teachable Machine, the result should be something similar to the image.



Left hand

Add Image Samples:

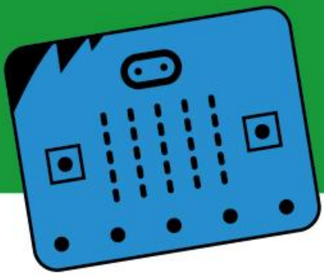
Right hand

Add Image Samples:

Non-raising hands

Add Image Samples:

Sheet 1



Tips for your project:

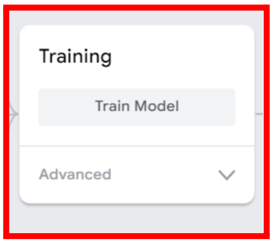
- The higher the number of images or samples, the better the Machine Learning model will work. However, the more images we add, the longer the training of the model will take.

- Another way to reduce the margin of error is to add several images from each category. For example, images of different people from different angles can be added, and so on. The more information the model provides, the better it will sort data.

- The model depends on the quality of the data entered. It is important that the image is well framed. It is also advisable not to include confusing images.

Stage 2:

Click on "TrainModel". The application will automatically validate the categories or classes entered.

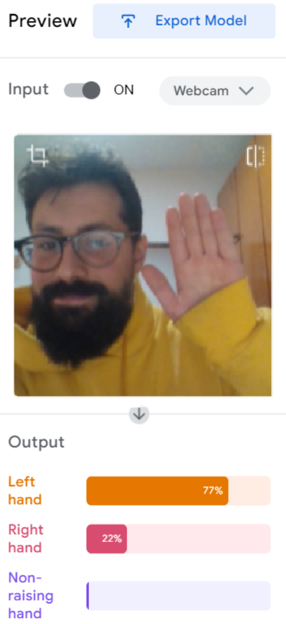


Stage 3:

In the "Preview" window, test the model and check its operation. For this you must activate the input device (in this case, the webcam) and test whether images are correctly sorted out.

Through the webcam, Teachable Machine receives information in image format.

In Preview mode, the "Output" box will show the matching percentage of what is being recorded by the camera with the different categories of the model created. This makes it possible to check that the model is working properly.

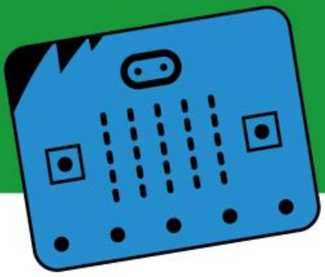


Input ON Webcam

Left hand 77%

Right hand 22%

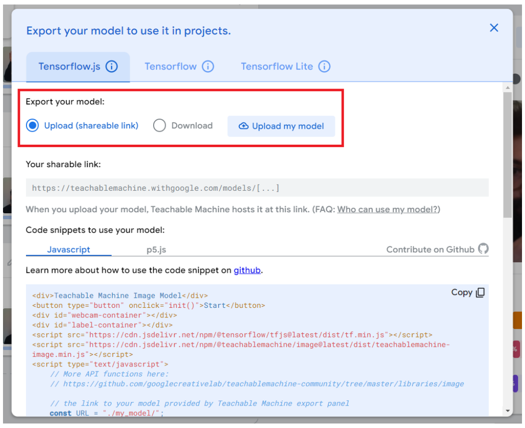
Non-raising hand



Stage 4:

a) Click on the “Export Model” button. The application uploads the model and generates an access link. Select the option “Upload (link to share)” and then click on “Upload My Model”.

b) Save this link; it will be later used in the projects.



Export your model:

● Upload (shareable link) ○ Download

Your sharable link:

```
https://teachablemachine.withgoogle.com/models/[...]
```

When you upload your model, Teachable Machine hosts it at this link. (FAQ: Who can use my model?)

Code snippets to use your model:

Learn more about how to use the code snippet on github.

```
<div>Teachable Machine Image Model</div>
<button type="button" onclick="init()">Start</button>
<div id="webcam-container"></div>
<div id="label-container"></div>
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/dist/teachablemachine-image.min.js"></script>
<script type="text/javascript">
    // More API functions here:
    // https://github.com/googlecreativelab/teachablemachine-community/tree/master/libraries/image

    // the link to your model provided by Teachable Machine export panel
    const URL = "./my_model/";
```

Tip:

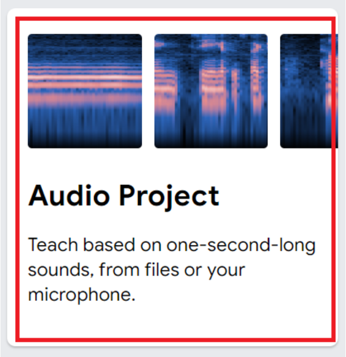
- Links generated by the application only contain the trained model. When the link is shared, only the model could be used. That is to say, you can test the model, but **cannot** modify the data with which it was trained.

4.2 Audio project

Stage 1:

Go to “New Project” and select “Audio Project”. Similarly to what happened with the webcam, when the data type is sound, Teachable Machine allows you to enter data through the microphone.

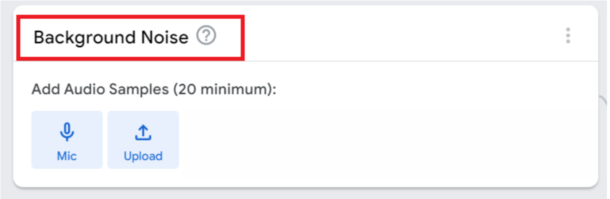
Sheet 1



Audio Project

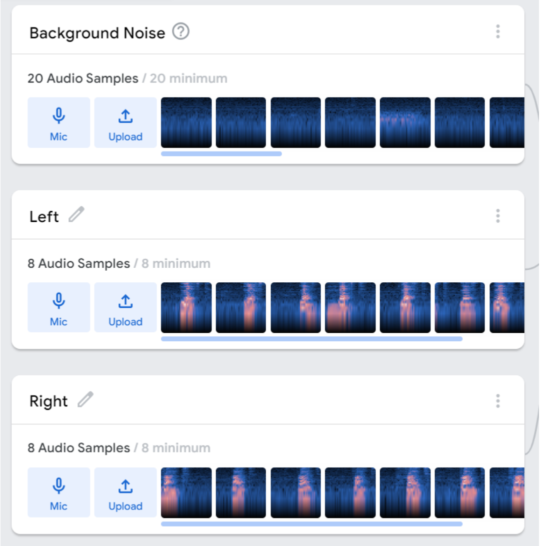
Teach based on one-second-long sounds, from files or your microphone.

IMPORTANT: In the case of audio, there is a preset class called **Background Noise**. Samples should be recorded in this category so that the model understands that there is a base noise and that the commands will be recorded with that sound in the background.



Background Noise

Add Audio Samples (20 minimum):



Background Noise

20 Audio Samples / 20 minimum

Left

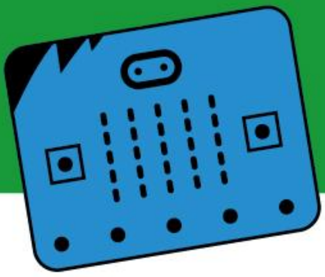
8 Audio Samples / 8 minimum

Right

8 Audio Samples / 8 minimum

Tips:

- In the "Background Noise" category, store at least 20 samples. For all other classes or categories, store at least 8 samples of each.
- In each class, modify the tones and the speed of the audios to generate a more robust model. This allows the model to understand different ways of saying the same data. For example, as in the case above, different ways of saying "right" and "left".



Stage 2:

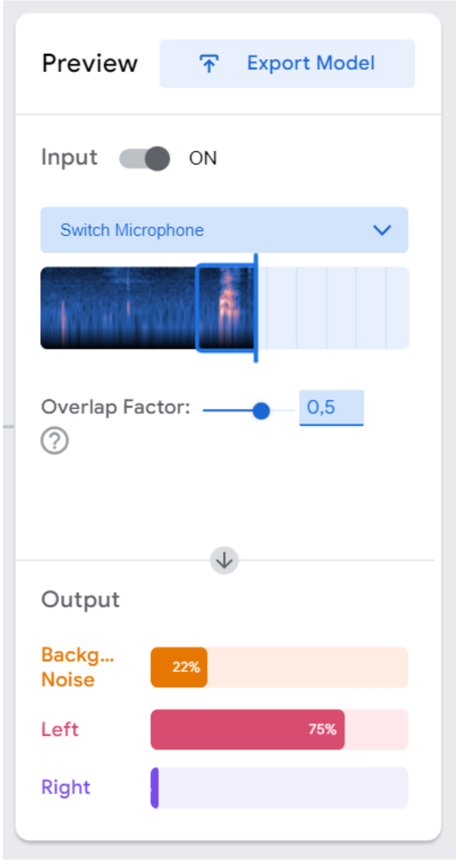
Preparing the model: click on this option once you have finished recording the different model classes or commands.

Stage 3:

a) Check model operation to evaluate how it works (evaluate the classes or the number and type of samples) in the “Preview” window.

b) Turn “On” the input device (in this case, the microphone) and test whether it correctly classifies the sounds (the words “left” and “right” in this example).

In case of modifications, the model can be reverified.



Preview

Export Model

Input ON

Switch Microphone

Overlap Factor: 0,5

Output

Backg... Noise 22%

Left 75%

Right

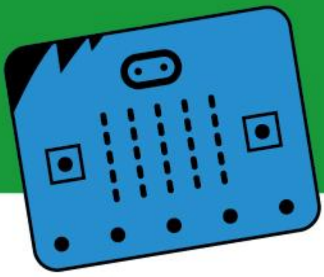
Stage 4:

a) Click on the “Export Model” button, as in the previous case. The application uploads the model and generates an access link. You have to select the option “Upload (link to share)” and then click on “Upload My Model”.

b) Save this link; it will be used later in the projects.

Tip:

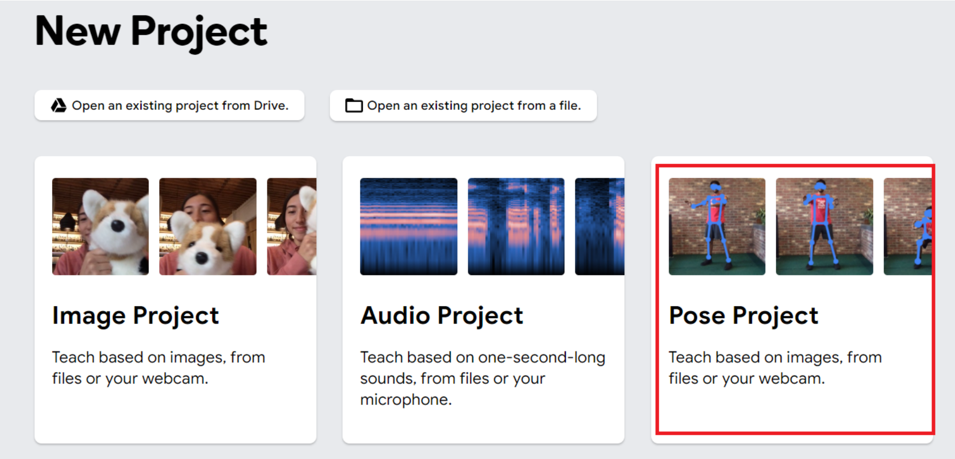
- Links generated by the application only contain the trained model. When the link is shared, the model can only be used. That is to say, you can test the model, but **cannot** modify the data with which it was trained.



4.3 Pose project:

Stage 1:

a) Select “Pose Project”.



New Project

Open an existing project from Drive.

Open an existing project from a file.

Image Project

Teach based on images, from files or your webcam.

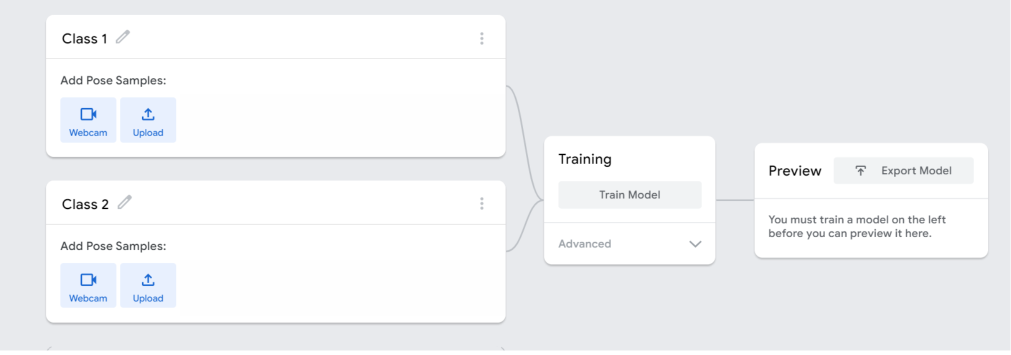
Audio Project

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

Pose Project

Teach based on images, from files or your webcam.

b) Enter and classify data.



Class 1

Add Pose Samples:

Webcam Upload

Class 2

Add Pose Samples:

Webcam Upload

Training

Train Model

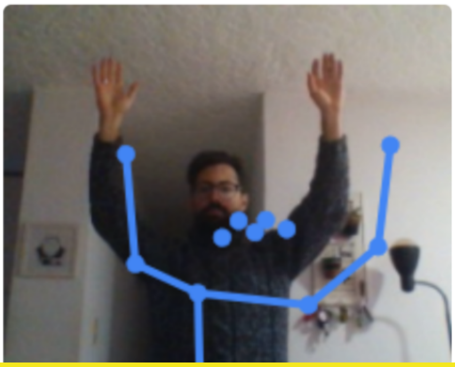
Advanced

Preview

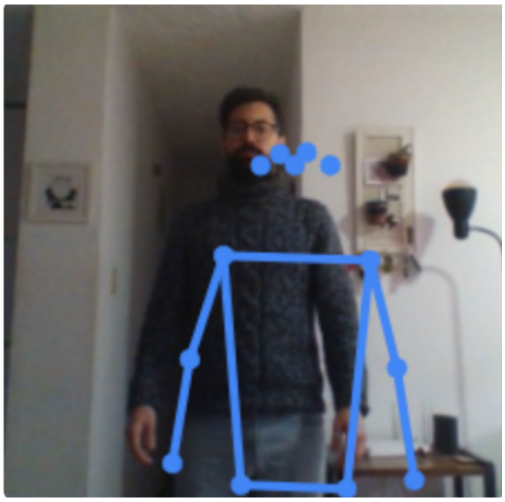
Export Model

You must train a model on the left before you can preview it here.

As an example, we can choose categories such as “Arms Up” and “Arms Down”. For the first one we should use the webcam to record poses in which the arms can be clearly seen extended and high. Important: we should check that points and lines



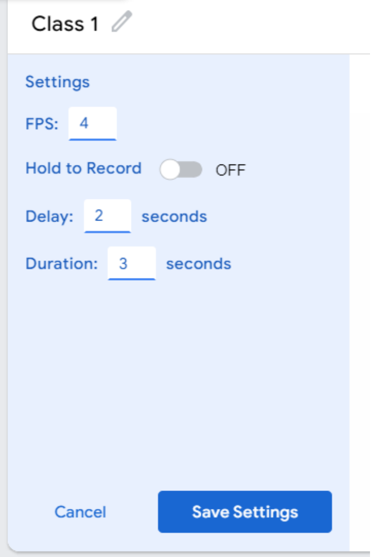
are defined in the image to recognize the pose, since that information is key for recognition.



For the next category we should use the webcam to record poses in which you can clearly see the arms down. In this case we should also check the use of lines and points to describe the pose we want to define.

The application allows users to add images from a camera or from the image database of your device (computer, cell phone, etc.).

If the webcam is used, data collection can be defined through set up. Tip: Take 4 FPS (frames per second) for 3 seconds (12 pieces of data in total). The delay is the time we have before you start recording, it is advisable to select 2 seconds. Those images that we do not consider representative of the class should be discarded.



Class 1

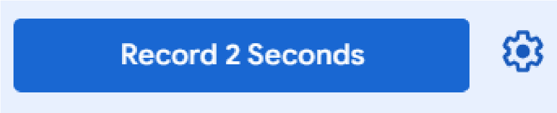
Settings

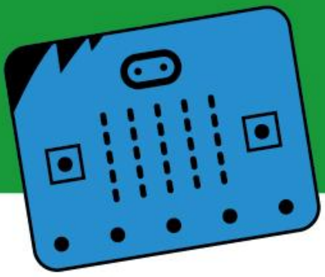
FPS: 4

Hold to Record ○ OFF

Delay: 2 seconds

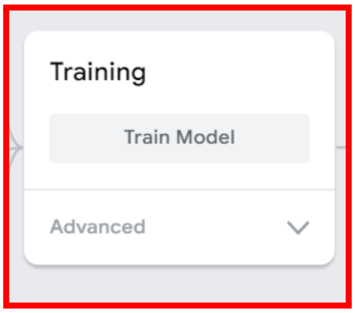
Duration: 3 seconds





Stage 2:

Click on “Train Model”. The application will automatically validate the categories or classes entered.

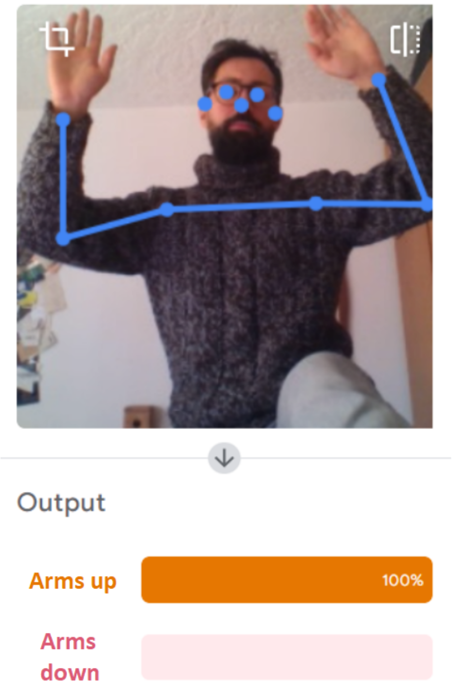


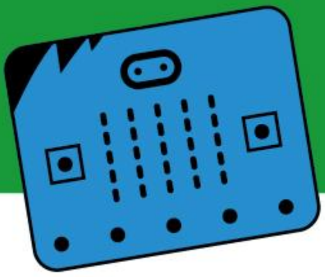
Stage 3:

In the “Preview” window, test the model and check its operation. For this you must activate the input device (in this case, the webcam) and test whether images are correctly sorted out.

Teachable Machine receives information in image format through the webcam

In “Preview Mode”, the “Output” box will show the matching percentage of what is being recorded with the different model categories. This makes it possible to check that the model is working properly.



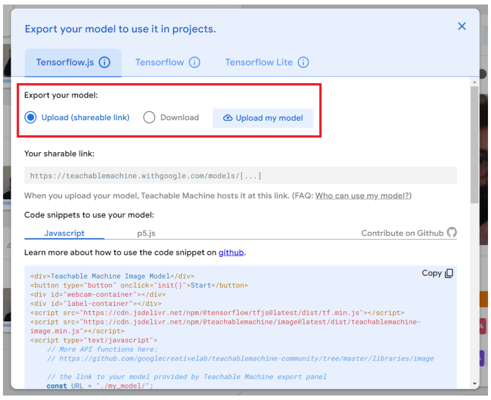


Stage 4:

a) Click on the “Export Model” button.

b) The application uploads the model and generates an access link. You have to select the option “Upload (link to share)” and then click on “Upload My Model”.

c) Save this link; it will be used later in the projects.



Export your model to use it in projects.

Tensorflow.js ⓘ | Tensorflow ⓘ | Tensorflow Lite ⓘ

Export your model:

● Upload (shareable link) ○ Download Upload my model

Your sharable link:

https://teachablemachine.withgoogle.com/models/[...]

When you upload your model, Teachable Machine hosts it at this link. (FAQ: Who can use my model?)

Code snippets to use your model:

Javascript | p5.js | Contribute on Github

Learn more about how to use the code snippet on github.

Copy

```
<div>Teachable Machine Image Model</div>
<button type="button" onclick="init()">Start</button>
<div id="webcam-container"></div>
<div id="label-container"></div>
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/dist/teachablemachine-
image.min.js"></script>
<script type="text/javascript">
    // More API functions here:
    // https://github.com/googlecreativelab/teachablemachine-community/tree/master/libraries/image

    // the link to your model provided by Teachable Machine export panel
    const URL = "./my_model/";
```

5 Preliminary considerations: What's next?

This sheet details the procedure to create Machine Learning models. In this case, simple models will identify, through images or audio samples, which hand the user raises.

However, the possible applications of this tool are countless. We invite you to continue testing Teachable Machine, adding more categories and generating new models.

In the following sheet we will explain how to link these models with a micro:bit board to expand the possibilities even more.

Don't miss it!