Chapter 3: Imagimob

After completing this chapter, you will understand how to use the machine learning solution from Imagimob, an Infineon Technologies company, to develop and train ML models for use with Infineon MCUs. You will work with solutions using inputs from motion and audio sensors.

Table of contents

Document conventions

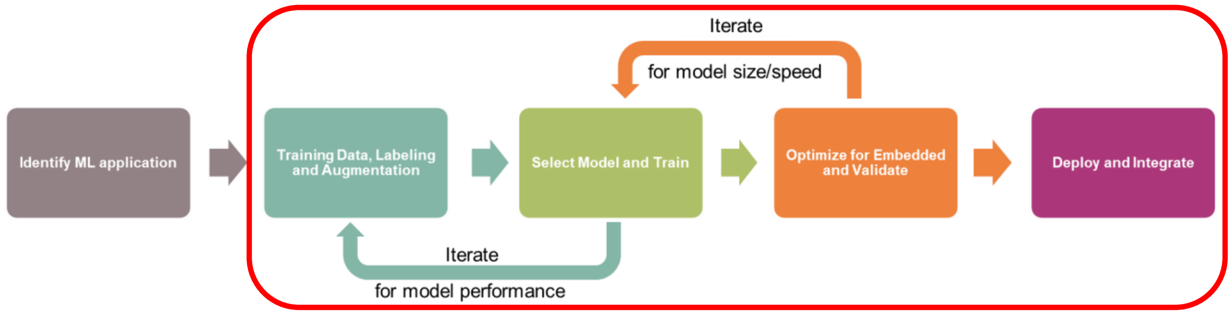
Convention	Usage	Example

`Courier New`	Displays code and text commands	`CY_ISR_PROTO(MyISR);`
		`make build`

Italics	Displays file names and paths	*sourcefile.hex*
[**bracketed, bold**]	Displays keyboard commands in procedures	[**Enter**] or [**Ctrl**] [**C**]
Menu > Selection	Represents menu paths	**File > New Project > Clone**
Bold	Displays GUI commands, menu paths and selections, and icon names in procedures	Click the **Debugger** icon, and then click **Next**.

3.1 Overview

In this chapter we will be using tools from Infineon's fully owned subsidiary Imagimob (https://imagimob.com/). Their tools help facilitate and automate the steps of capturing raw data from a sensor, labeling the data, generating, and training a machine learning model, validating the model on the target hardware, and then optimizing it and validating the optimized model on the target hardware.



Imagimob can output C files that can be directly deployed and integrated on the MCU. However, it can also output a Keras *.h5* model that can be used by the ModusToolbox™ ML configurator for additional optimization such as quantization. The ModusToolbox™ ML configurator will be discussed in the next chapter.

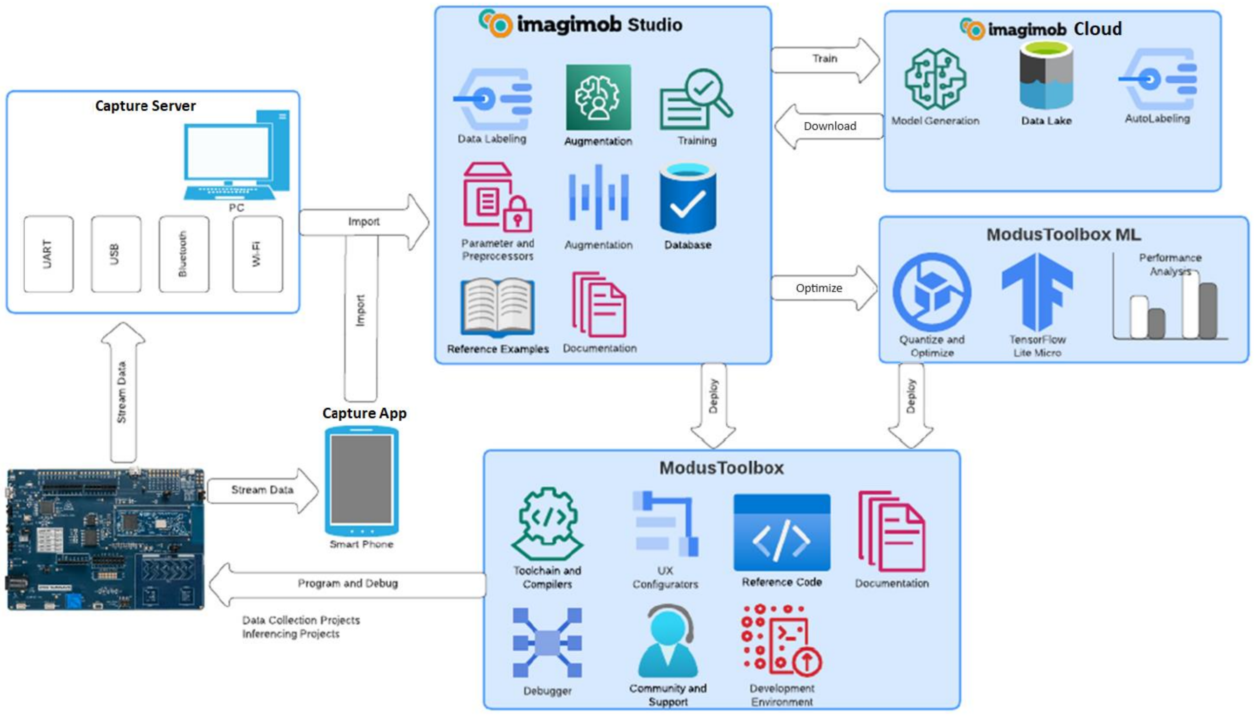
The Imagimob AI is designed to provide solutions to a wide range of problems including:

- **Predictive maintenance -** Recognize machine state, detect machine anomalies, and act in milliseconds.
- **Audio applications -** Classify sound events, spot keywords, and recognize your sound environment.
- **Gesture recognition -** Detect hand gestures using low-power radars, capacitive touch sensors or accelerometers.
- **Signal classification -** Recognize repeatable signal patterns from any sensor .
- **Fall detection -** Fall detection using IMUs or a single accelerometer.
- **Material detection -** Real time material detection using low-power radars.

The Imagimob AI software consists of the Imagimob Cloud, Imagimob Studio, the Capture App, and the Capture Server.

Data from the PSoC™ 6 and its attached sensor(s) is captured and relayed to either the Capture App or the Capture Server. These tools associate the collected data with a label file and an optional video showing exactly what was done. When the data set is ready, Imagimob Studio is used to collect and manage data files and define expectations of the model. When the model is defined, the data is all uploaded to Imagimob Cloud which generates different models and trains them. Results are then presented as downloadable options via Imagimob Studio again.

The models and code generated by Imagimob Studio are fully reusable on any device. The ModusToolbox™ ML Configurator can optionally be used to optimize the model generated by Imagimob Studio for optimal performance on the PSoC™ 6 hardware.



Note: *You can read more about the Imagimob integration into the ModusToolbox™ ecosystem here. One of the links on that page is to Application Note AN238663, which contains additional details about the ModusToolbox™ Imagimob flow.*

3.2 Workflow

Now we'll go into more detail about the workflow in Imagimob and the skills required at each step.

3.2.1 Create an Imagimob project

In Imagimob studio, you can create an empty classification project, create a starter project that is close to your needs, or import an existing project. The starter projects include labeled data sets, pre-processing configurations, and pre-trained models, but you can add/remove data and retrain for your own purposes.

3.2.2 Collect data

As you learned earlier, getting a good data set is one of the largest factors in developing a successful machine learning solution. With Imagimob, the Capture App or Capture Server can be used to quickly collect data directly from your sensor(s) of choice and label it. As you will see, the videos captured along with the data make it easy to tell exactly what was happening. This facilitates creating precise labels for each point in time and helps with future reviews of the data.

A ModusToolbox™ code example (Machine Learning Imagimob Data Collection) allows you to program the PSoC™ 6 device to capture data from audio or motion sensors without requiring you to write any firmware. Options in the *Makefile* allow you to specify the method used to send the data – Bluetooth® LE, USB, UART and Wi-Fi are supported.

Data collection from additional sensors such as pressure and radar sensors will be added in the future.

The Python-based Capture Server desktop tool and the Android based Capture App can be used interchangeably. Both tools serve the same purpose and the choice of which to use ultimately comes down to which makes the data collector's job easier. In some cases, it may even make sense to collect some data with one tool and other data with the other tool. The Capture Server can support receiving data over Bluetooth® LE, USB, UART, or Wi-Fi connections and uses an attached webcam to capture video. The Capture App supports receiving data over Bluetooth® LE or Wi-Fi and uses the phone's built-in camera. Either way, once data is collected, it must be transferred to a PC for use with Imagimob Studio.

The person(s) responsible for capturing data will need some expertise in the end application. That is, they will need to understand how the sensor(s) will be used in the real world and what the valid range of inputs looks like to ensure that the data set contains sufficient information to create a good ML model.

Once you have finished collecting data, it must be added to Imagimob Studio, and you can move on to managing and analyzing the data.

Note: *In the first release, only UART is supported for capturing data.*

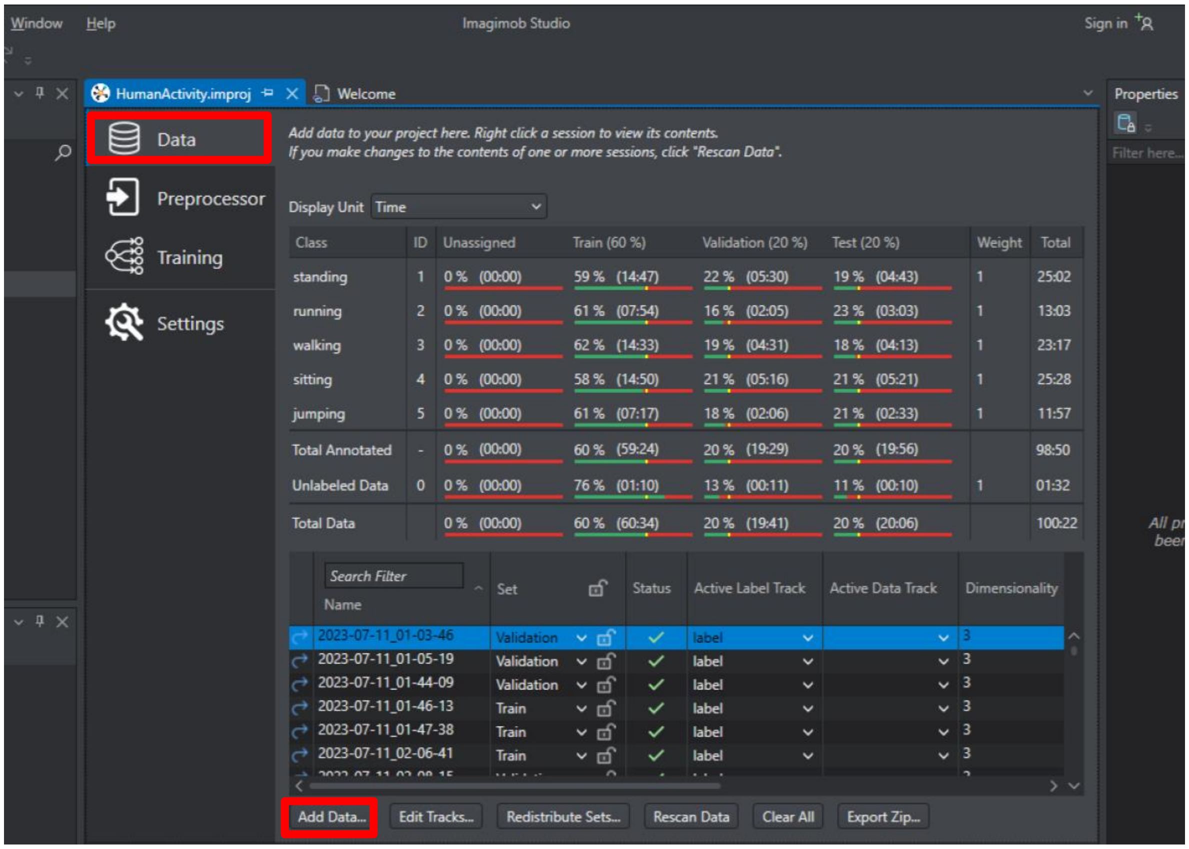
Note: *The easiest way to learn how to use the capture server is to create the Machine Learning Imagimob Data Collection code example and open the README.md file.*

Note: *Online data collection is also available using Graph UX. Refer to the following link for details: https://developer.imagimob.com/data-collection/collect-data-using-graph-ux*

3.2.3 Import, analyze, and label data

Once data has been collected, it needs to be added to the project within Imagimob Studio. The first step is to move the new data files into the *Data* directory in the project. Each data set that you collected should be in its own directory under the *Data* directory. That's how the capture server organizes the data, so you should be able to just move the directories.

Once the data is in the project directory, open the project file (*.improj*) from Imagimob studio, select the **Data** tab, and then click the **Add Data** button at the bottom of the screen.



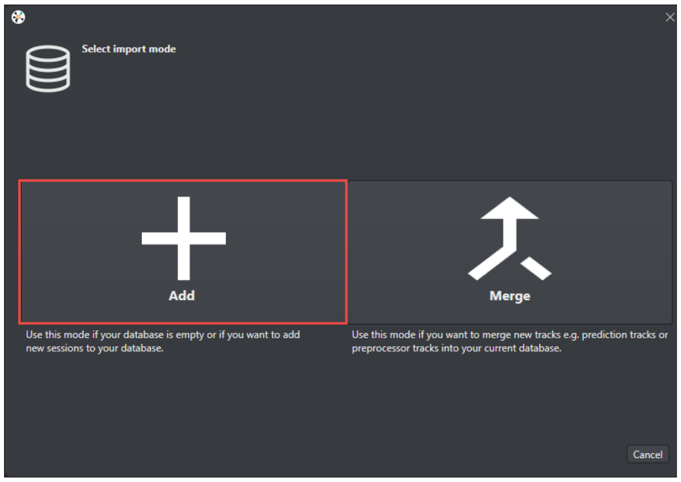
Add data to your project here. Right click a session to view its contents.
If you make changes to the contents of one or more sessions, click "Rescan Data".

Display Unit: Time

Class	ID	Unassigned	Train (60 %)	Validation (20 %)	Test (20 %)	Weight	Total
standing	1	0 % (00:00)	59 % (14:47)	22 % (05:30)	19 % (04:43)	1	25:02
running	2	0 % (00:00)	61 % (07:54)	16 % (02:05)	23 % (03:03)	1	13:03
walking	3	0 % (00:00)	62 % (14:33)	19 % (04:31)	18 % (04:13)	1	23:17
sitting	4	0 % (00:00)	58 % (14:50)	21 % (05:16)	21 % (05:21)	1	25:28
jumping	5	0 % (00:00)	61 % (07:17)	18 % (02:06)	21 % (02:33)	1	11:57
Total Annotated	-	0 % (00:00)	60 % (59:24)	20 % (19:29)	20 % (19:56)		98:50
Unlabeled Data	0	0 % (00:00)	76 % (01:10)	13 % (00:11)	11 % (00:10)	1	01:32
Total Data		0 % (00:00)	60 % (60:34)	20 % (19:41)	20 % (20:06)		100:22

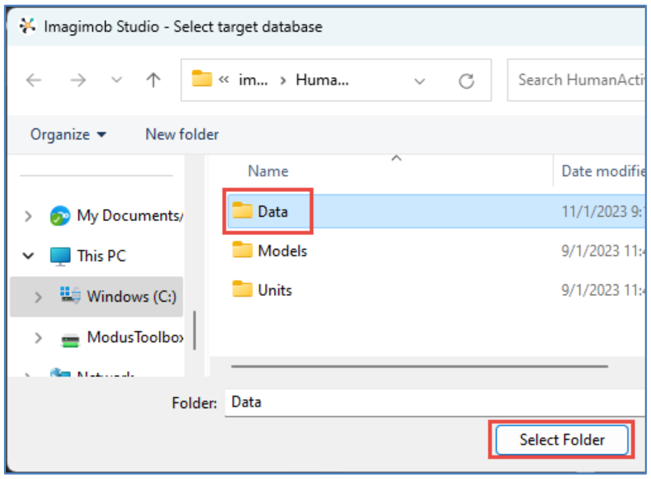
Name	Set	Status	Active Label Track	Active Data Track	Dimensionality
2023-07-11_01-03-46	Validation	✓	label		3
2023-07-11_01-05-19	Validation	✓	label		3
2023-07-11_01-44-09	Validation	✓	label		3
2023-07-11_01-46-13	Train	✓	label		3
2023-07-11_01-47-38	Train	✓	label		3
2023-07-11_02-06-41	Train	✓	label		3

Click the **Add** button in the Select import mode window, navigate to, and select the *Data* directory from the project and click **Select Folder**.

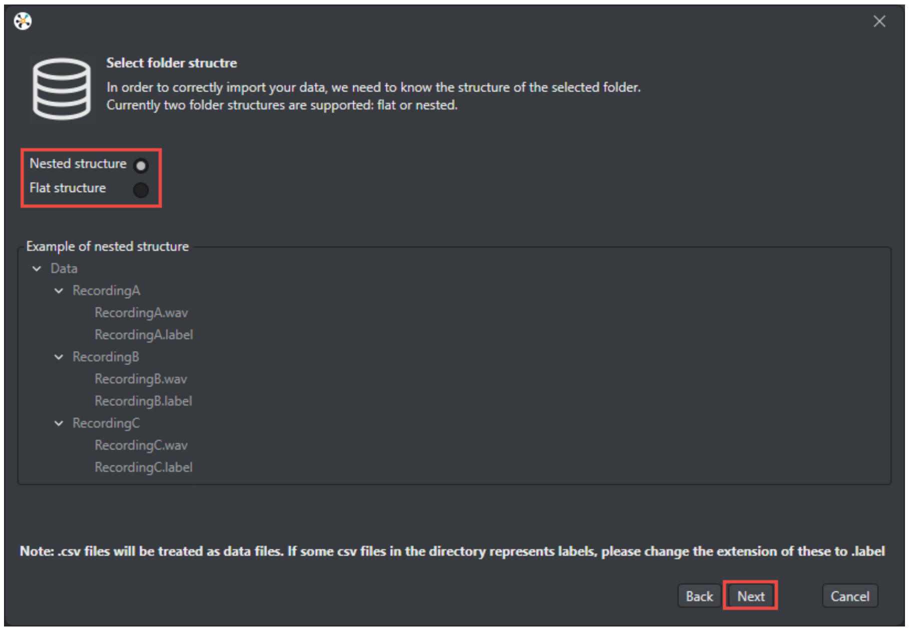


Use this mode if your database is empty or if you want to add new sessions to your database.

Use this mode if you want to merge new tracks e.g. prediction tracks or preprocessor tracks into your current database.



Select **Nested structure** and click **Next**. You can then select the data sets you want to add. If you leave everything selected and click **OK**, you will be asked what you want to do with repeated data sets. Click **Don't Replace** to add only the new data to the project.



Select folder structre

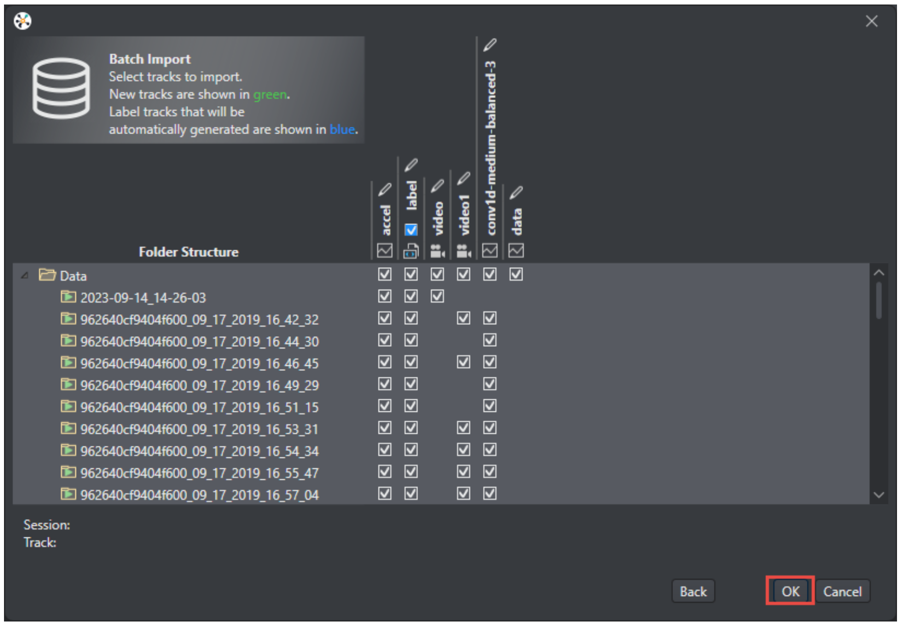
In order to correctly import your data, we need to know the structure of the selected folder.
Currently two folder structures are supported: flat or nested.

Nested structure [✓]
Flat structure

Example of nested structure

```
Data
  RecordingA
    RecordingA.wav
    RecordingA.label
  RecordingB
    RecordingB.wav
    RecordingB.label
  RecordingC
    RecordingC.wav
    RecordingC.label
```

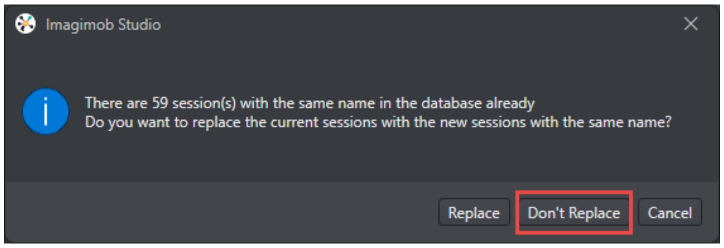
Note: .csv files will be treated as data files. If some csv files in the directory represents labels, please change the extension of these to .label



Batch Import
Select tracks to import.
New tracks are shown in green.
Label tracks that will be automatically generated are shown in blue.

Folder Structure	accel	label	video	video1	conv1d-medium-balanced-3	data
Data	✓	✓	✓	✓	✓	✓
2023-09-14_14-26-03	✓	✓	✓			
962640cf9404f600_09_17_2019_16_42_32	✓	✓		✓	✓	
962640cf9404f600_09_17_2019_16_44_30	✓	✓			✓	
962640cf9404f600_09_17_2019_16_46_45	✓	✓		✓	✓	
962640cf9404f600_09_17_2019_16_49_29	✓	✓			✓	
962640cf9404f600_09_17_2019_16_51_15	✓	✓			✓	
962640cf9404f600_09_17_2019_16_53_31	✓	✓		✓	✓	
962640cf9404f600_09_17_2019_16_54_34	✓	✓		✓	✓	
962640cf9404f600_09_17_2019_16_55_47	✓	✓		✓	✓	
962640cf9404f600_09_17_2019_16_57_04	✓	✓		✓	✓	

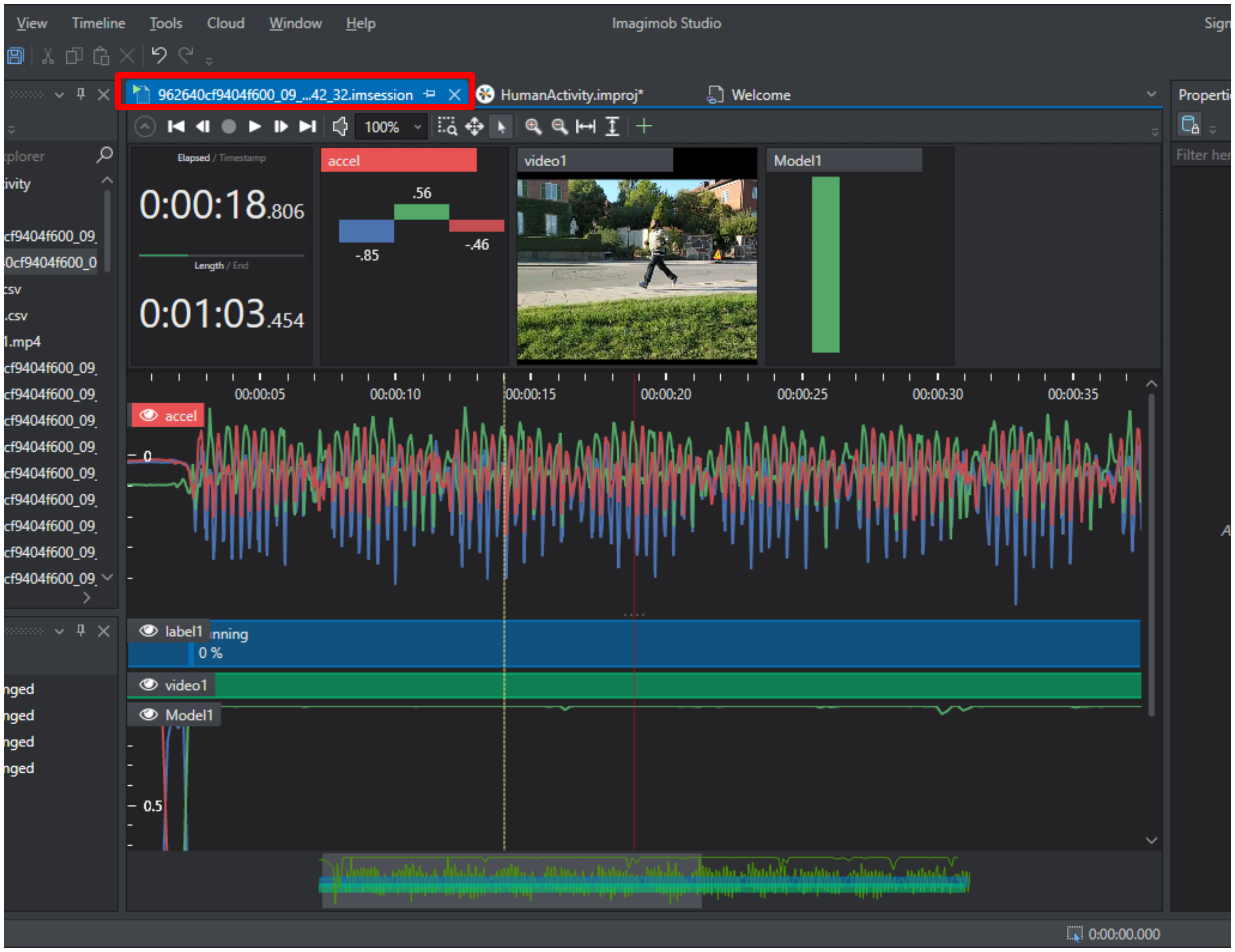
Session:
Track:



Imagimob Studio

There are 59 session(s) with the same name in the database already
Do you want to replace the current sessions with the new sessions with the same name?

The new data will appear in the list of data sets at the bottom of the Data window. Each data set (captured data, label, and video) is considered its own session and gets a *.imsession* file associated with it. These session files can be opened by double clicking on the data set name in the table. This will open a tab for the session that looks like the image below:



You can use the session window to review and label the data and ensure the video, data, and labels are all aligned with each other.

Note: All of the data not associated with a label will be considered as belonging to the "unlabeled" class. All classification projects will have the "unlabeled" class in addition to the user-defined classes as output of the model(s).

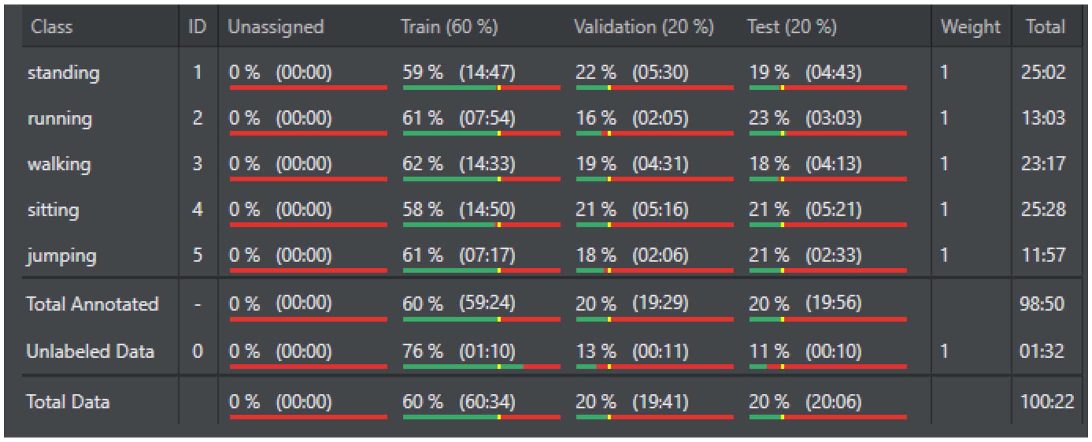
The play head is always located at the center of the window. You can zoom in and out using the buttons at the top. When you are zoomed in, you can move the track back and forth using a left click and drag in the grey box at the bottom or using a right click and drag in the area where the tracks are shown.

To add a label, left click and drag in the track area to select a region. Then, right click in the label track and select **New Label**. Once a label is created, you can move it or drag the ends to align it with the data.

Note: *If you have an existing model, you can import it as an evaluation model and then use it to label data sets for you. This can be used to auto-label new data sets.*

When the data all looks good, and you have a sufficiently robust data set to start trying out models, return to the project configuration, and **Rescan** for changes to the data.

The columns in the table at the top of the Data window show details about the data, so that you can see if you have enough data for each classification and if it is distributed properly.



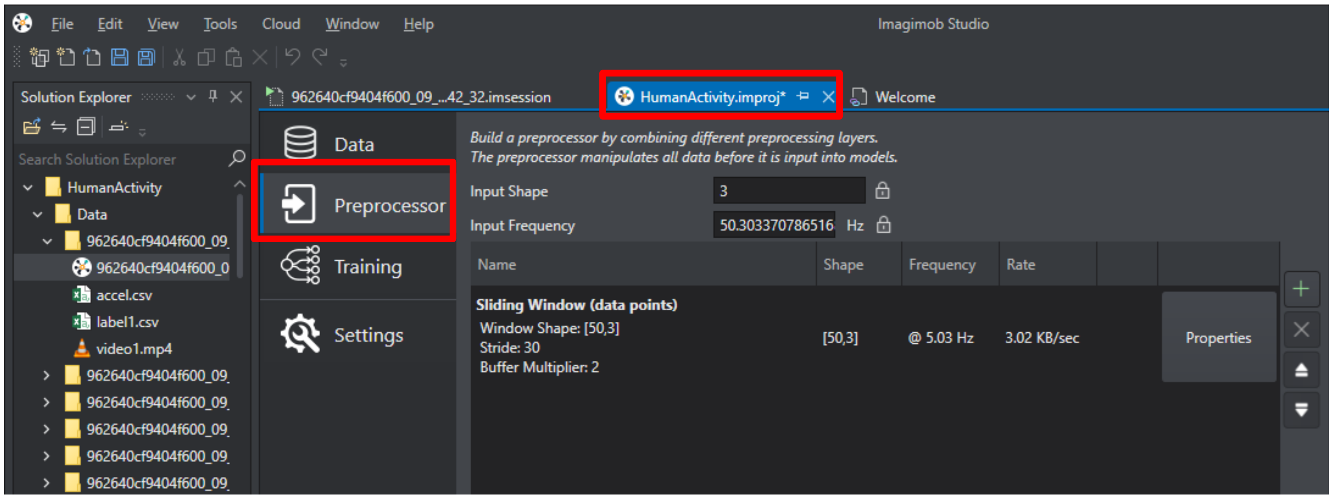
Class	ID	Unassigned	Train (60 %)	Validation (20 %)	Test (20 %)	Weight	Total
standing	1	0 % (00:00)	59 % (14:47)	22 % (05:30)	19 % (04:43)	1	25:02
running	2	0 % (00:00)	61 % (07:54)	16 % (02:05)	23 % (03:03)	1	13:03
walking	3	0 % (00:00)	62 % (14:33)	19 % (04:31)	18 % (04:13)	1	23:17
sitting	4	0 % (00:00)	58 % (14:50)	21 % (05:16)	21 % (05:21)	1	25:28
jumping	5	0 % (00:00)	61 % (07:17)	18 % (02:06)	21 % (02:33)	1	11:57
Total Annotated	-	0 % (00:00)	60 % (59:24)	20 % (19:29)	20 % (19:56)		98:50
Unlabeled Data	0	0 % (00:00)	76 % (01:10)	13 % (00:11)	11 % (00:10)	1	01:32
Total Data		0 % (00:00)	60 % (60:34)	20 % (19:41)	20 % (20:06)		100:22

Each data set can be assigned to be used as Training, Validation, or Test data. If you click the **Redistribute Sets** button, the tool can automatically assign the data sets based on the percentages that you choose for each type of data. You can also manually assign data sets using the drop-down menus in the **Set** column. Once you manually assign a data set, it will be locked so that the automatic redistribution will not change it unless you unlock it first.

3.2.4 Build and train models

Once the data set is complete, it's time to create and train a model. For these steps you will continue to use Imagimob Studio, but will also rely on the Imagimob Cloud for the computationally intense training.

The first step is to define any pre-processing operations necessary. This can reshape the data before it is passed into the model and is thus important to do first so the input layer can be created with the right shape. Some preprocessing operations that might be considered are to create a Sliding Window (e.g. for repetitive motion data), FFT Shift, or some type of smoothing function (e.g. for jittery motion data).

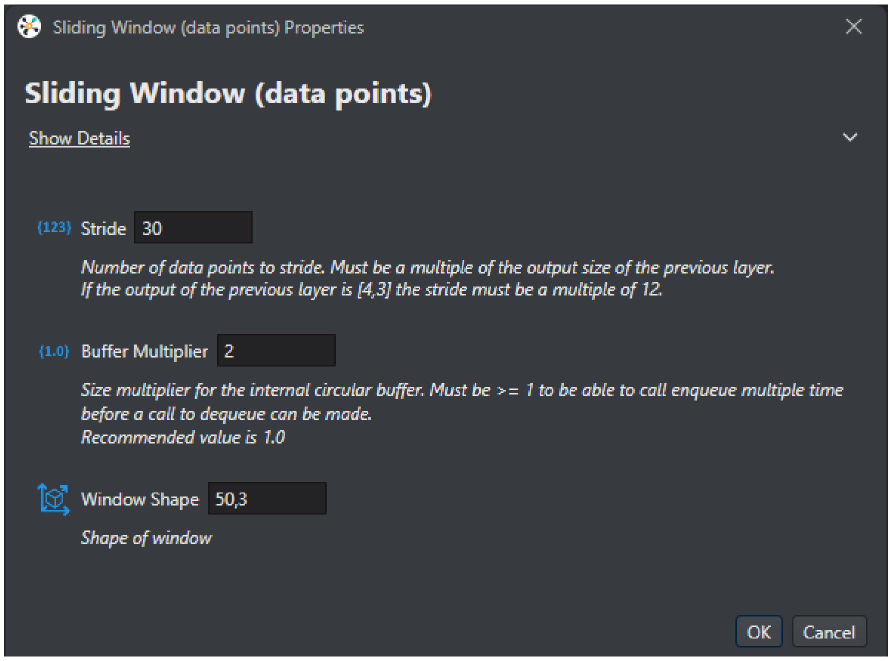


Build a preprocessor by combining different preprocessing layers.
The preprocessor manipulates all data before it is input into models.

Input Shape: 3
Input Frequency: 50.303370786516 Hz

Name	Shape	Frequency	Rate
Sliding Window (data points) Window Shape: [50,3] Stride: 30 Buffer Multiplier: 2	[50,3]	@ 5.03 Hz	3.02 KB/sec

As an example, the preprocessing for the Human Activity IMU example uses a sliding window that looks like this:



Sliding Window (data points)

Stride: 30

Number of data points to stride. Must be a multiple of the output size of the previous layer.
If the output of the previous layer is [4,3] the stride must be a multiple of 12.

Buffer Multiplier: 2

Size multiplier for the internal circular buffer. Must be >= 1 to be able to call enqueue multiple time before a call to dequeue can be made.
Recommended value is 1.0

Window Shape: 50,3

Shape of window

The input layer to the sliding window is the raw IMU data which has a shape of 3 (X, Y, and Z motion values). The stride is 30 which means that the window slides across 10 samples at a time (10 X, Y, Z samples). Since the input is samples at 50 Hz, a stride across 10 samples means that the windows slides 200ms for each step (50 Hz / 10 samples = 5 Hz = 200ms).

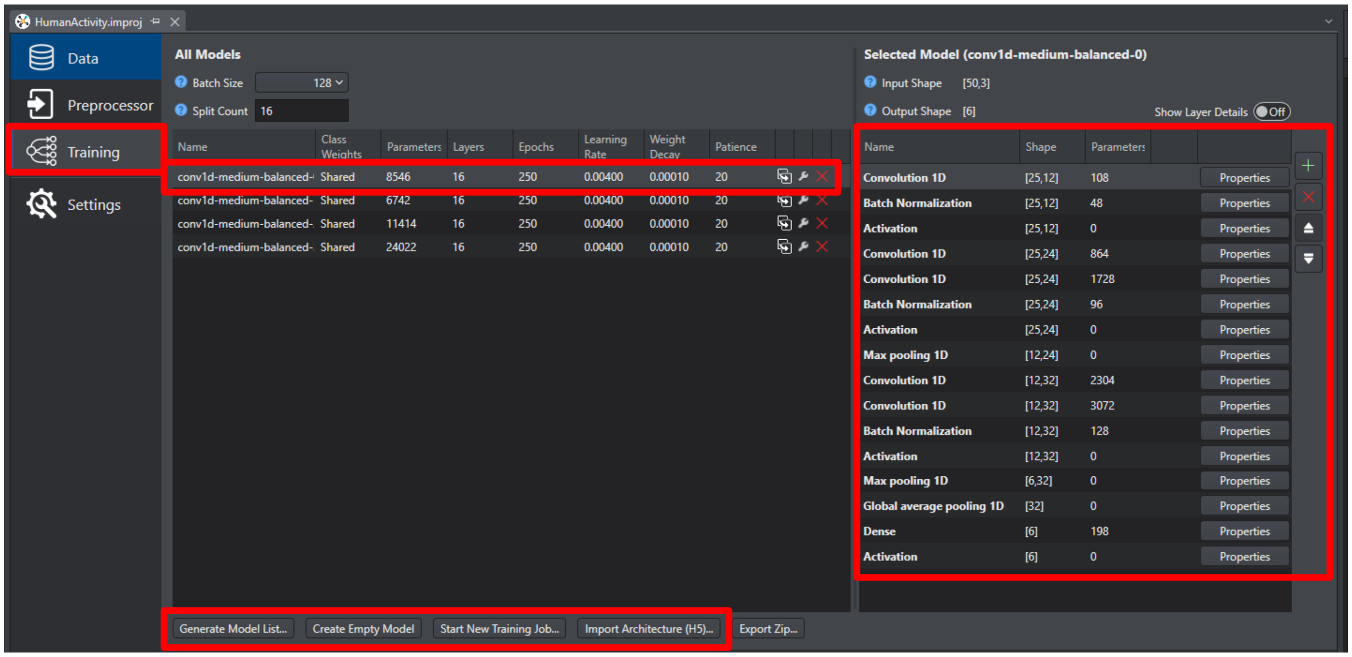
The output window shape is 50,3. This means that each window sent to the model will be 50 samples worth of IMU X, Y, Z data. Since the input sampling frequency is 50 Hz, this means that each window is 1 second in duration.

Once you have defined the preprocessor settings, you can click on **Create Track from Preprocessor…** which will open up the wizard to create a track to allow you to see the preprocessor data in the *.imsession* files.

After applying any preprocessor transformations to the data, it is ready to be fed into the training process. For this, switch to the **Training** tab.

By clicking **Generate Model List**, the tool can automatically create a set of models (full auto-ML). These models will have different layers and layer configurations and thus will produce slightly different results. You are always free to create a custom model from scratch by clicking **Create Empty Model** or you can clone one of the auto-ML models as a starting point for a new model.

For each model in the list, you can change training settings such as the number of epochs to be run and the learning rate used by entering the desired values. Click the wrench icon to set additional model properties. If you select a model, you can see the layers that make up that model. You can re-order the layers, change each layer's properties, and even add/remove layers before training.



All Models

Batch Size 128

Split Count 16

Name	Class Weights	Parameters	Layers	Epochs	Learning Rate	Weight Decay	Patience
conv1d-medium-balanced-	Shared	8546	16	250	0.00400	0.00010	20
conv1d-medium-balanced-	Shared	6742	16	250	0.00400	0.00010	20
conv1d-medium-balanced-	Shared	11414	16	250	0.00400	0.00010	20
conv1d-medium-balanced-	Shared	24022	16	250	0.00400	0.00010	20

Selected Model (conv1d-medium-balanced-0)

Input Shape [50,3]

Output Shape [6]

Show Layer Details Off

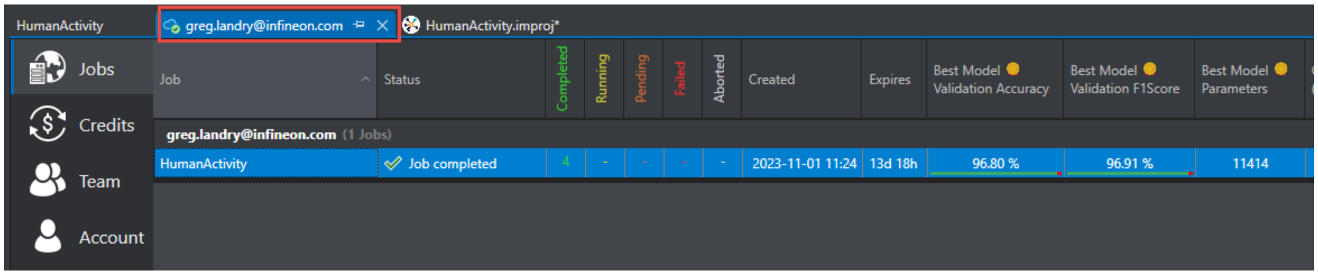
Name	Shape	Parameters	
Convolution 1D	[25,12]	108	Properties
Batch Normalization	[25,12]	48	Properties
Activation	[25,12]	0	Properties
Convolution 1D	[25,24]	864	Properties
Convolution 1D	[25,24]	1728	Properties
Batch Normalization	[25,24]	96	Properties
Activation	[25,24]	0	Properties
Max pooling 1D	[12,24]	0	Properties
Convolution 1D	[12,32]	2304	Properties
Convolution 1D	[12,32]	3072	Properties
Batch Normalization	[12,32]	128	Properties
Activation	[12,32]	0	Properties
Max pooling 1D	[6,32]	0	Properties
Global average pooling 1D	[32]	0	Properties
Dense	[6]	198	Properties
Activation	[6]	0	Properties

Note: *You can import an existing Keras H5 model in different modes using the **Import Architecture (H5)** button. The Training mode allows full training of the model while Evaluation only allows you to use the model to generate predictions from the data sets. Other options allow limited re-training. The Evaluation mode may be used to help generate labels for your data sets.*

Once you have a list of models, they can be trained and evaluated to see which one produces the best results. The training is started by clicking the **Start New Training Job**. At this point you can decide if you want to use GPUs for the training or just the default CPUs. The GPU option will use more credits but will typically result in faster training. Once you start the job, it will upload all the data and model information to the Imagimob Cloud to perform the actual training. As jobs finish, you can review the performance.

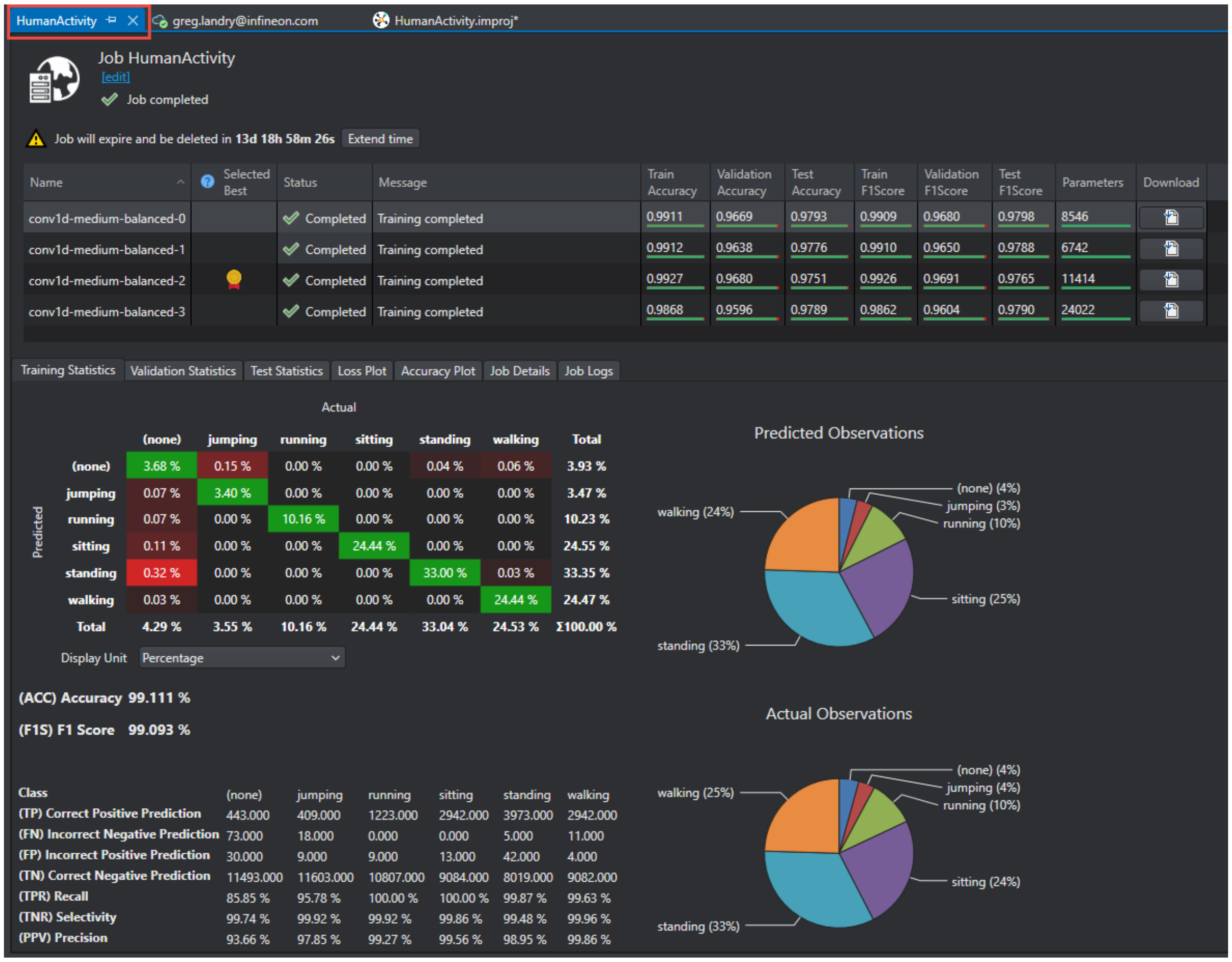
3.2.5 Evaluate the models

The training progress can be monitored by opening the cloud connection – the button to do this is in the upper right corner in Imagimob Studio. You must login first for the button to be active. This shows all the projects that you have created and provides a summary of the training status for each project.



Job	Status	Completed	Running	Pending	Failed	Aborted	Created	Expires	Best Model Validation Accuracy	Best Model Validation F1Score	Best Model Parameters
greg.landry@infineon.com (1 Jobs)											
HumanActivity	Job completed	4	-	-	-	-	2023-11-01 11:24	13d 18h	96.80 %	96.91 %	11414

Double clicking on any project will open it to view the result details as shown below. In this case, four models have been trained.



Name	Selected Best	Status	Message	Train Accuracy	Validation Accuracy	Test Accuracy	Train F1Score	Validation F1Score	Test F1Score	Parameters	Download
conv1d-medium-balanced-0		Completed	Training completed	0.9911	0.9669	0.9793	0.9909	0.9680	0.9798	8546	
conv1d-medium-balanced-1		Completed	Training completed	0.9912	0.9638	0.9776	0.9910	0.9650	0.9788	6742	
conv1d-medium-balanced-2	●	Completed	Training completed	0.9927	0.9680	0.9751	0.9926	0.9691	0.9765	11414	
conv1d-medium-balanced-3		Completed	Training completed	0.9868	0.9596	0.9789	0.9862	0.9604	0.9790	24022	

Predicted \ Actual	(none)	jumping	running	sitting	standing	walking	Total
(none)	3.68 %	0.15 %	0.00 %	0.00 %	0.04 %	0.06 %	3.93 %
jumping	0.07 %	3.40 %	0.00 %	0.00 %	0.00 %	0.00 %	3.47 %
running	0.07 %	0.00 %	10.16 %	0.00 %	0.00 %	0.00 %	10.23 %
sitting	0.11 %	0.00 %	0.00 %	24.44 %	0.00 %	0.00 %	24.55 %
standing	0.32 %	0.00 %	0.00 %	0.00 %	33.00 %	0.03 %	33.35 %
walking	0.03 %	0.00 %	0.00 %	0.00 %	0.00 %	24.44 %	24.47 %
Total	4.29 %	3.55 %	10.16 %	24.44 %	33.04 %	24.53 %	Σ100.00 %

Display Unit: Percentage

(ACC) Accuracy 99.111 %

(F1S) F1 Score 99.093 %

Class	(none)	jumping	running	sitting	standing	walking
(TP) Correct Positive Prediction	443.000	409.000	1223.000	2942.000	3973.000	2942.000
(FN) Incorrect Negative Prediction	73.000	18.000	0.000	0.000	5.000	11.000
(FP) Incorrect Positive Prediction	30.000	9.000	9.000	13.000	42.000	4.000
(TN) Correct Negative Prediction	11493.000	11603.000	10807.000	9084.000	8019.000	9082.000
(TPR) Recall	85.85 %	95.78 %	100.00 %	100.00 %	99.87 %	99.63 %
(TNR) Selectivity	99.74 %	99.92 %	99.92 %	99.86 %	99.48 %	99.96 %
(PPV) Precision	93.66 %	97.85 %	99.27 %	99.56 %	98.95 %	99.86 %

After the training jobs have completed running, performance data is available for each model. This includes the accuracy and F1 score for each phase (Train, Validate, Test). It also provides details about the parameters/complexity of the model. You can look at the table and charts that show how well the model distinguishes between data points, how well it performs against the test data, what features are used, what layers are used, etc. The tool will recommend the model that it thinks performed best.

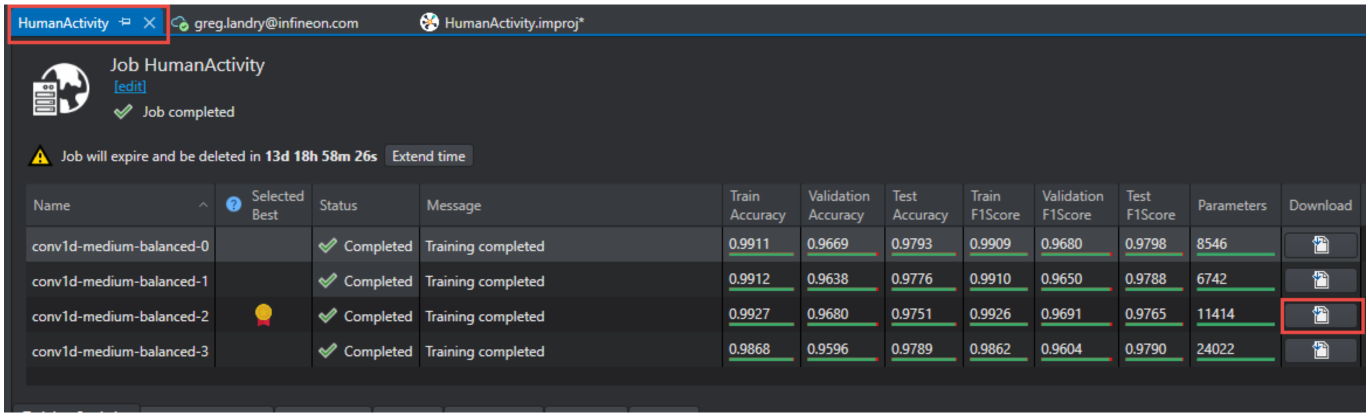
3.2.6 Download the best model

Depending on the overall satisfaction with the results provided by the training there are a few options:

- Gather more data.
- Adjust some of the data.
- Modify the pre-processor options.
- Create or modify one or more models.
- Accept one of the models.
- Use one of the models as input to the ModusToolbox™ ML configurator to perform additional optimization.

For any of the first four options, simply go to the prior step, make adjustments at that point, and then proceed through again. The rest of this section will focus on moving forward with a selected model or as an input to the ModusToolbox™ ML configurator.

To download a model, click on the icon in the **Download** column for the model you choose to proceed with.

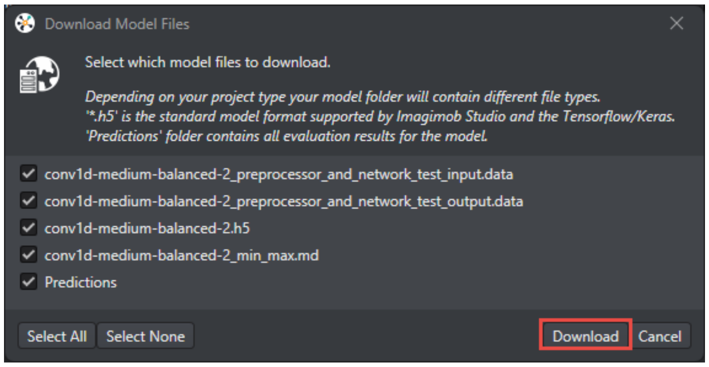


Job completed

Job will expire and be deleted in **13d 18h 58m 26s**

Name	Selected Best	Status	Message	Train Accuracy	Validation Accuracy	Test Accuracy	Train F1Score	Validation F1Score	Test F1Score	Parameters	Download
conv1d-medium-balanced-0		✓ Completed	Training completed	0.9911	0.9669	0.9793	0.9909	0.9680	0.9798	8546	
conv1d-medium-balanced-1		✓ Completed	Training completed	0.9912	0.9638	0.9776	0.9910	0.9650	0.9788	6742	
conv1d-medium-balanced-2	●	✓ Completed	Training completed	0.9927	0.9680	0.9751	0.9926	0.9691	0.9765	11414	
conv1d-medium-balanced-3		✓ Completed	Training completed	0.9868	0.9596	0.9789	0.9862	0.9604	0.9790	24022	

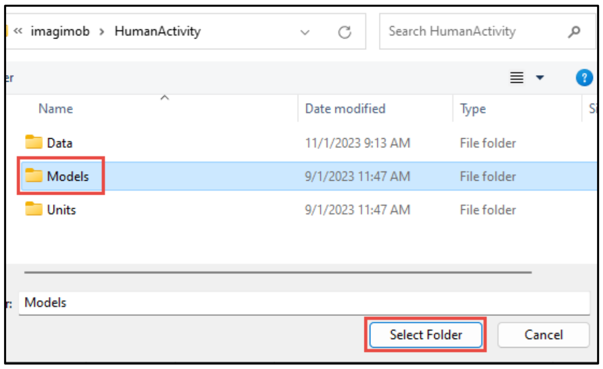
In the **Download Model Files** window, click **Download** and then navigate to the *Models* folder inside the project. Click **Select Folder** to begin the download process.



Select which model files to download.

Depending on your project type your model folder will contain different file types.
'.h5' is the standard model format supported by Imagimob Studio and the Tensorflow/Keras.*
'Predictions' folder contains all evaluation results for the model.

- ✓ conv1d-medium-balanced-2_preprocessor_and_network_test_input.data
- ✓ conv1d-medium-balanced-2_preprocessor_and_network_test_output.data
- ✓ conv1d-medium-balanced-2.h5
- ✓ conv1d-medium-balanced-2_min_max.md
- ✓ Predictions

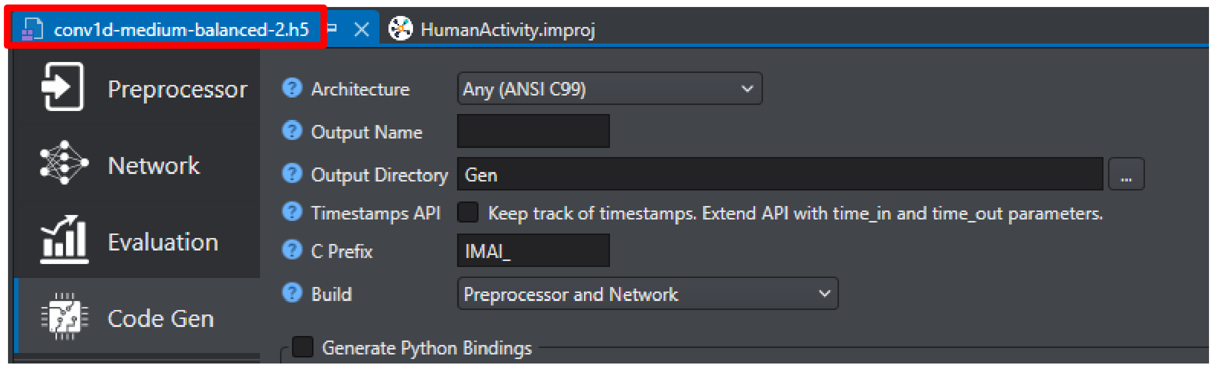


Name	Date modified	Type
Data	11/1/2023 9:13 AM	File folder
Models	9/1/2023 11:47 AM	File folder
Units	9/1/2023 11:47 AM	File folder

Models

After downloading, the model can be opened in Imagimob Studio to view its details. Just double-click on the *.h5* file in the *Models/<model-name>* directory to open it. This view provides four tabs of interest:

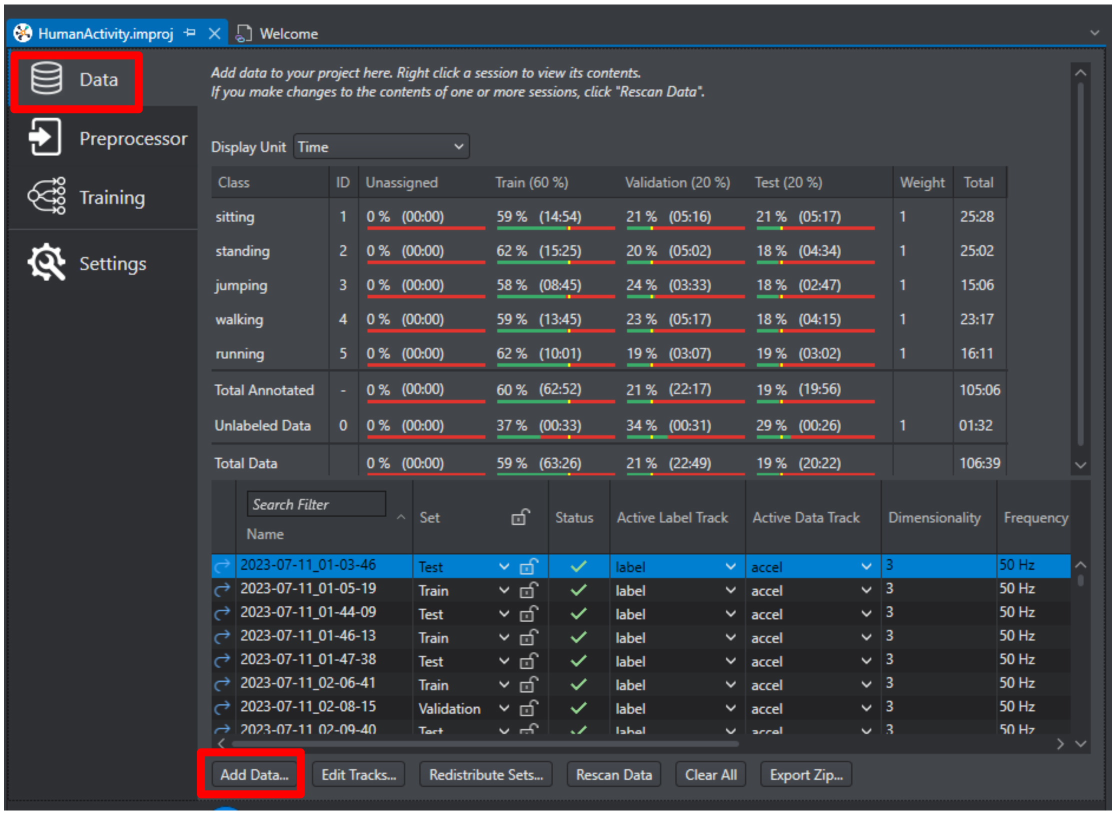
- Preprocessor: This is the same as the preprocessor information defined for the project.
- Network: This provides details about the model's layers, activations and parameters.
- Evaluation: This is the same performance data as was available from the cloud training but with additional filtering and visualization capabilities.
- Code Gen: This allows you to define details about how the model should be deployed to the edge device and then generate the code.



Architecture: Any (ANSI C99)
Output Name:
Output Directory: Gen
Timestamps API: Keep track of timestamps. Extend API with time_in and time_out parameters.
C Prefix: IMAI_
Build: Preprocessor and Network
Generate Python Bindings

3.2.7 Compare the model predictions to the collected data

One of the items downloaded along with the Keras *.h5* model is a set of predictions. These can be used to evaluate the trained model by comparing the predictions the model makes vs. the original labels in the data sets. In Imagimob Studio, we can get a much more detailed view of the performance of a model by merging the model predictions as tracks into our sessions containing the original data. You can do this by using the data import tool of your project. Open the *.improj* file and navigate to the **Data** tab, then click the **Add Data** button.



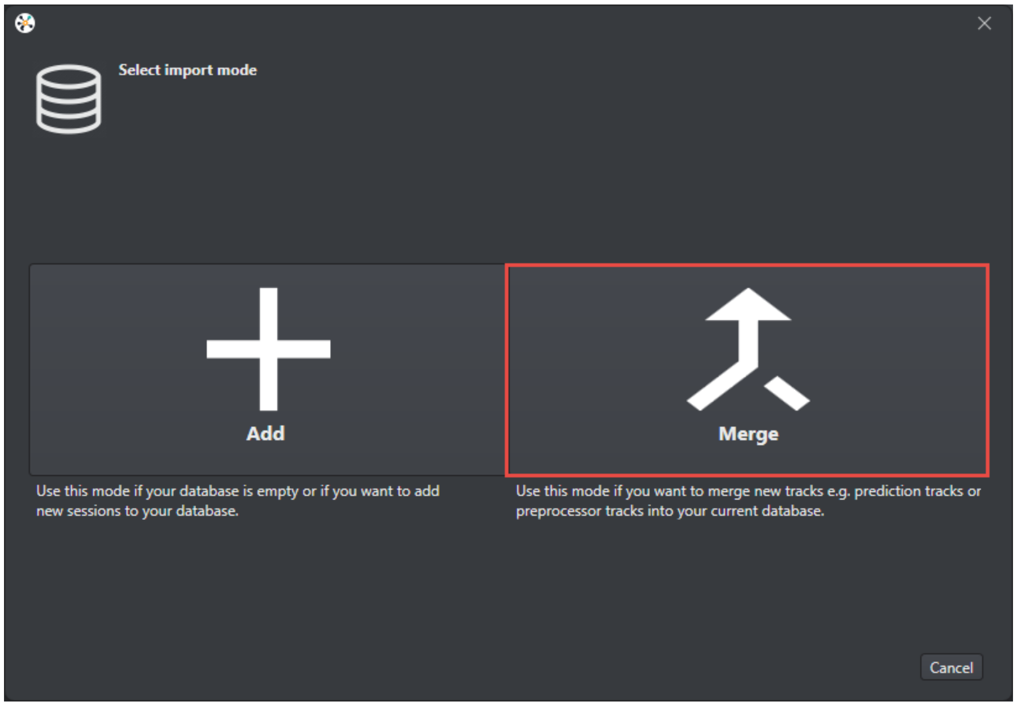
Add data to your project here. Right click a session to view its contents.
If you make changes to the contents of one or more sessions, click "Rescan Data".

Display Unit: Time

Class	ID	Unassigned	Train (60 %)	Validation (20 %)	Test (20 %)	Weight	Total
sitting	1	0 % (00:00)	59 % (14:54)	21 % (05:16)	21 % (05:17)	1	25:28
standing	2	0 % (00:00)	62 % (15:25)	20 % (05:02)	18 % (04:34)	1	25:02
jumping	3	0 % (00:00)	58 % (08:45)	24 % (03:33)	18 % (02:47)	1	15:06
walking	4	0 % (00:00)	59 % (13:45)	23 % (05:17)	18 % (04:15)	1	23:17
running	5	0 % (00:00)	62 % (10:01)	19 % (03:07)	19 % (03:02)	1	16:11
Total Annotated	-	0 % (00:00)	60 % (62:52)	21 % (22:17)	19 % (19:56)		105:06
Unlabeled Data	0	0 % (00:00)	37 % (00:33)	34 % (00:31)	29 % (00:26)	1	01:32
Total Data		0 % (00:00)	59 % (63:26)	21 % (22:49)	19 % (20:22)		106:39

Name	Set	Status	Active Label Track	Active Data Track	Dimensionality	Frequency
2023-07-11_01-03-46	Test	✓	label	accel	3	50 Hz
2023-07-11_01-05-19	Train	✓	label	accel	3	50 Hz
2023-07-11_01-44-09	Test	✓	label	accel	3	50 Hz
2023-07-11_01-46-13	Train	✓	label	accel	3	50 Hz
2023-07-11_01-47-38	Test	✓	label	accel	3	50 Hz
2023-07-11_02-06-41	Train	✓	label	accel	3	50 Hz
2023-07-11_02-08-15	Validation	✓	label	accel	3	50 Hz
2023-07-11_02-09-40	Test	✓	label	accel	3	50 Hz

In the dialog, select to **Merge** the prediction data with the existing data sets.



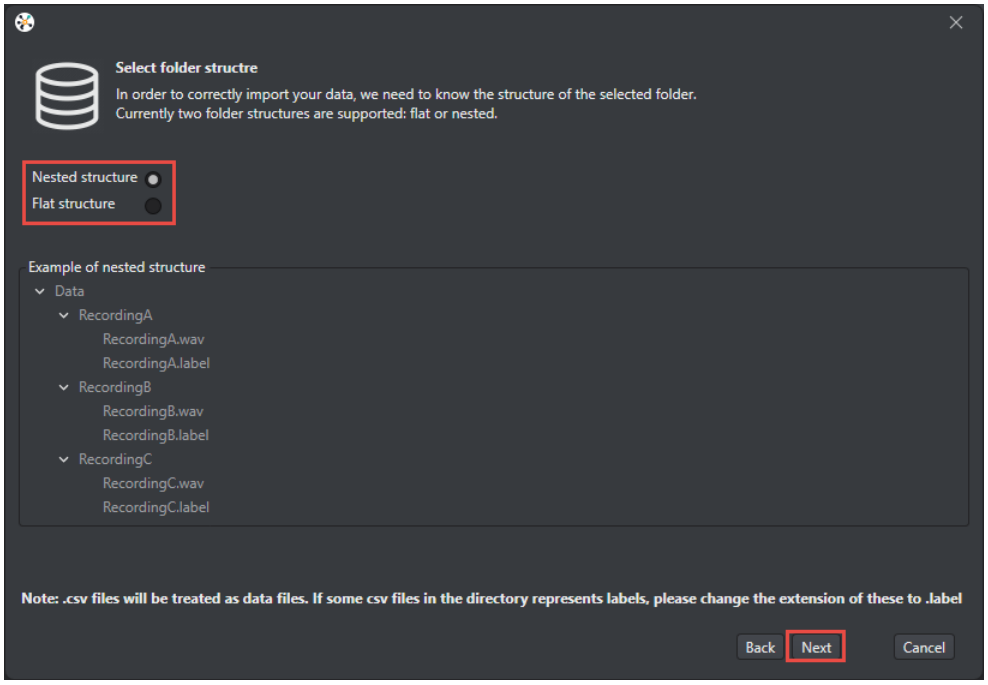
Select import mode

Use this mode if your database is empty or if you want to add new sessions to your database.

Use this mode if you want to merge new tracks e.g. prediction tracks or preprocessor tracks into your current database.

This will then bring up a navigation window to pick what data to merge in. For this, navigate to the Predictions folder for the model that was downloaded above. This will be in the project's Model directory. Once selected, click **Select Folder**.

In the subsequent dialog, select the appropriate structure of the prediction folder, typically it has the "Nested" structure and click **Next**.



Select folder structre

In order to correctly import your data, we need to know the structure of the selected folder. Currently two folder structures are supported: flat or nested.

Nested structure ●
Flat structure

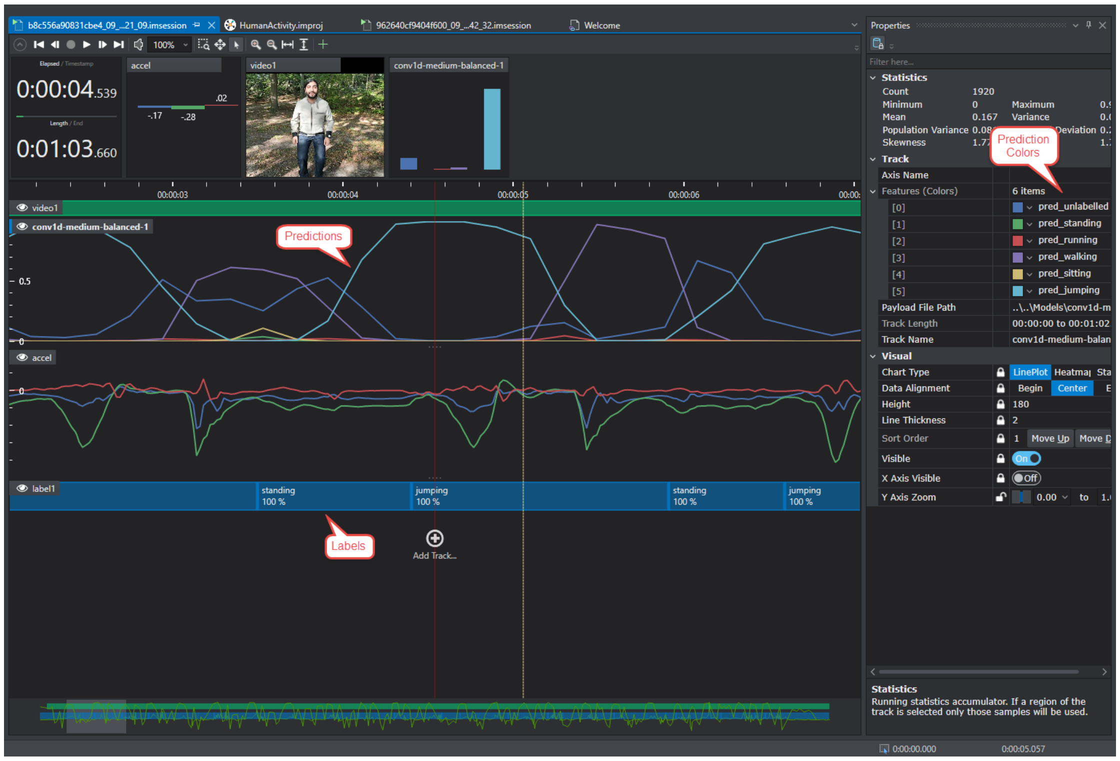
Example of nested structure

```
Data
    RecordingA
        RecordingA.wav
        RecordingA.label
    RecordingB
        RecordingB.wav
        RecordingB.label
    RecordingC
        RecordingC.wav
        RecordingC.label
```

Note: .csv files will be treated as data files. If some csv files in the directory represents labels, please change the extension of these to .label

The predictions from the *Predictions* folder we downloaded will be merged as tracks into the corresponding local sessions. In the dialog asking to select data and label tracks, click **Next**.

Now it's finally time to see how the imported model predictions compare to the labels from the training data. Double-click a session in the table of the **Data** tab of your project to open.



Statistics			
Count	1920		
Minimum	0	Maximum	0.9
Mean	0.167	Variance	0.0
Population Variance	0.08	Deviation	0.2
Skewness	1.77		1.1

Track	
Axis Name	
Features (Colors)	6 items
[0]	🟦 pred_unlabelled
[1]	🟩 pred_standing
[2]	🟥 pred_running
[3]	🟪 pred_walking
[4]	🟨 pred_sitting
[5]	🟦 pred_jumping
Payload File Path	..\..\Models\conv1d-m
Track Length	00:00:00 to 00:01:02
Track Name	conv1d-medium-balan
Visual	
Chart Type	[✓ LinePlot] Heatmap Sta
Data Alignment	Begin [✓ Center] E
Height	180
Line Thickness	2
Sort Order	1 Move Up Move D
Visible	On
X Axis Visible	Off
Y Axis Zoom	0.00 to 1.

Statistics
Running statistics accumulator. If a region of the track is selected only those samples will be used.

0:00:00.000 0:00:05.057

In the main window, there is a combined plot of the labels, the model predictions (here "conv1d_medium_balanced_1"), and the original data. The model predictions are shown as solid lines with different colors depending on the class. The properties window shows what color maps to what prediction. This allows for easy comparison of the three to determine if there are any discrepancies between the label and predictions. If so, the data can be looked at to see if there are anomalies or if the label is possibly incorrect. If the label and data both seem correct, it may indicate that the model was not trained on sufficient data to be as robust as needed.

You can merge in predictions from more than one model at a time (assuming you downloaded more than one model) if you want to compare how different models perform. You can also hide some data tracks to analyze other tracks more easily.

You can also use the model evaluation tools in Imagimob Studio to jump to misclassified samples in the data sets. For more information on how to do this, see https://developer.imagimob.com/model-building/model-evaluation-using-confusion-matrix#model-evaluation-using-confusion-matrix.

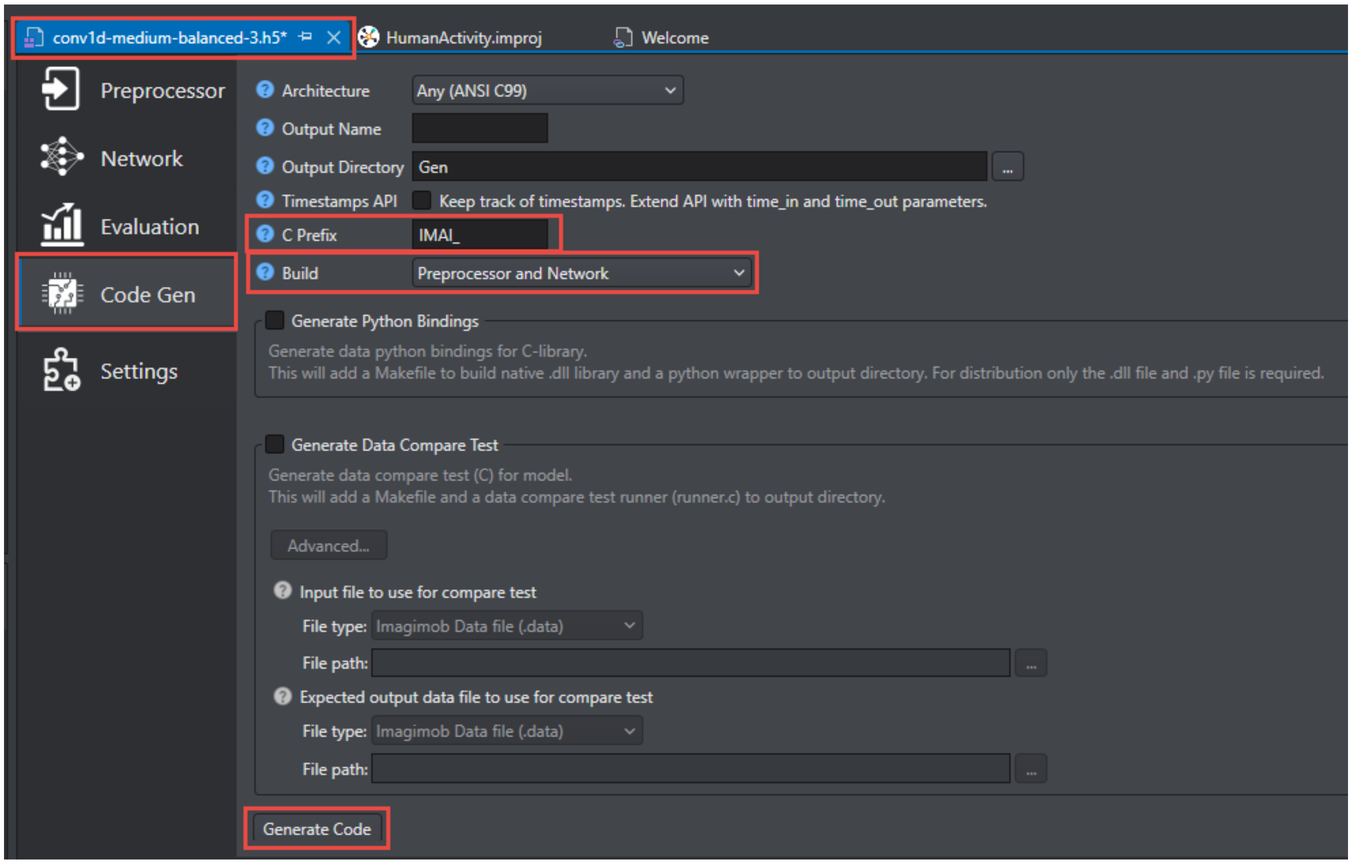
3.2.8 Test on the device

The final step to verify the model is to test on the target device to make sure it behaves as expected. Imagimob models are fully self-contained and do not require the ModusToolbox™ Machine Learning Configurator, but the tool can be used to perform quantization and optimization if desired. The following sections cover each case: using the Imagimob model directly or importing the model into ModusToolbox™ ML configurator to do additional optimization.

3.2.8.1 Use Imagimob model directly

If you want to use the Imagimob model directly, go to the tab containing the Keras *.h5* model and open the **Code Gen** tab to generate a C model that can be used in your firmware. By selecting **Preprocessor and Network** you get *.c* and *.h* files that include any data preprocessing that you selected, such as a sliding window and the model itself.

You can also select **Preprocessor and Quantized Network** if you want the model to be quantized for more efficient processing. With that option, you can specify how much accuracy can be tolerated by quantization. Click **Generate Code** once you select the desired settings.



Architecture: Any (ANSI C99)
Output Name:
Output Directory: Gen
Timestamps API: Keep track of timestamps. Extend API with time_in and time_out parameters.
C Prefix: IMAI_
Build: Preprocessor and Network

Generate Python Bindings
Generate data python bindings for C-library.
This will add a Makefile to build native .dll library and a python wrapper to output directory. For distribution only the .dll file and .py file is required.

Generate Data Compare Test
Generate data compare test (C) for model.
This will add a Makefile and a data compare test runner (runner.c) to output directory.

Input file to use for compare test
File type: Imagimob Data file (.data)
File path:
Expected output data file to use for compare test
File type: Imagimob Data file (.data)
File path:

The Machine Learning Imagimob Deploy (*mtb-example-ml-imagimob-deploy*) code example is available to drop in your model for testing. The code example assumes that the **C prefix** for the models is **IMAI_** as shown above.

Once you have the *.c* and *.h* files from Imagimob Studio, copy them to the *source* directory of the deploy application and replace the *model.c* and *model.h* files. In addition to the model itself, the *model.c* file contains comments with detailed information about the model.

The API to use the model consists of three main functions:

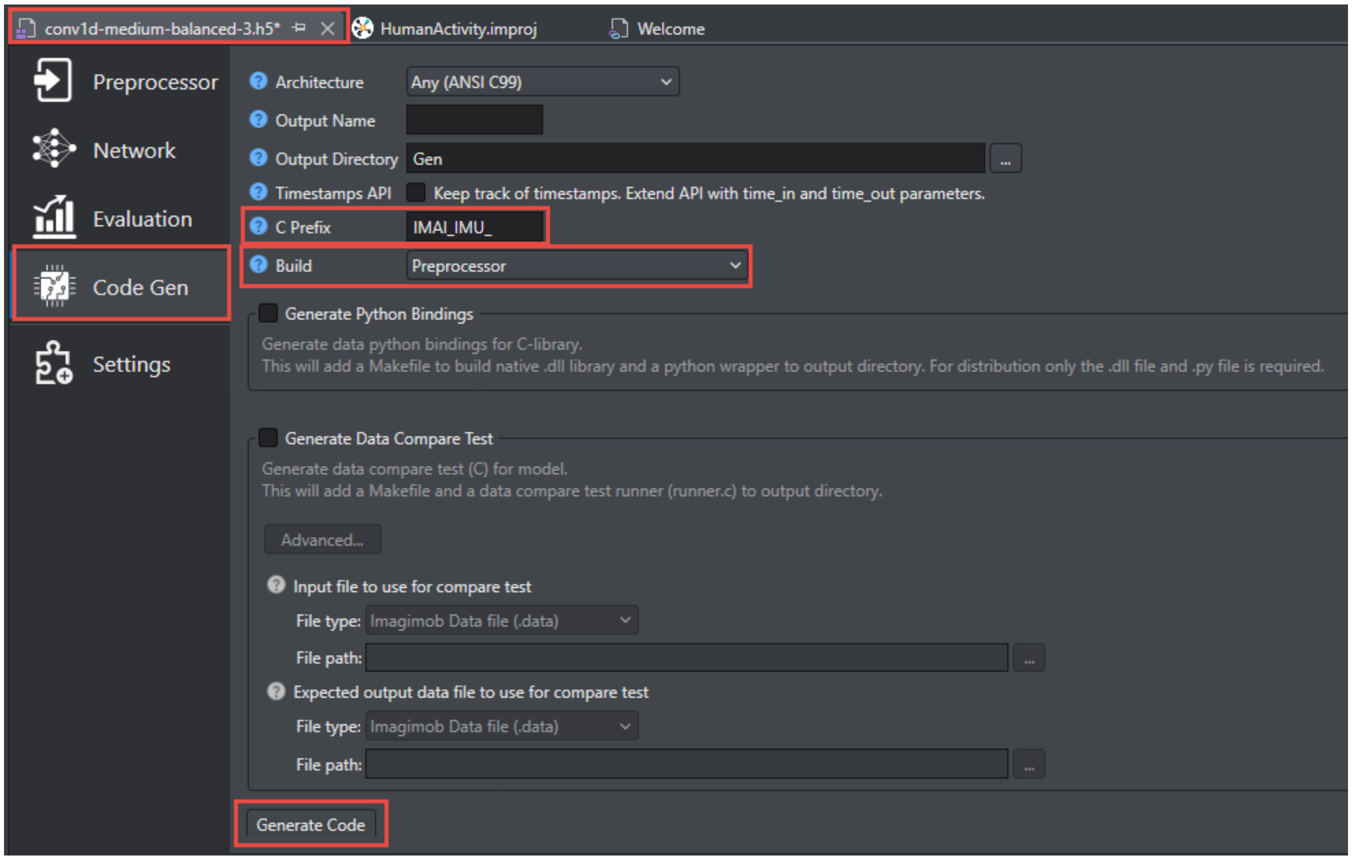
`IMAI_init`	This function is called once to initialize the model.
`IMAI_enqueue`	This function is called each time a new sample is ready to be sent to the preprocessor and the model. It is non-blocking.
`IMAI_dequeue`	This function is called to get results from the model. The return value must be checked to determine if a new result is available since the enqueue function is non-blocking.

3.2.8.2 Use ModusToolbox™ ML configurator to optimize the model

If you want to try out further optimization and quantization using the ModusToolbox™ ML Configurator, you can import the Keras *.h5* model from Imagimob into the ModusToolbox™ ML Configurator and optimize it.

Note: Using the ModusToolbox™ ML Configurator will be covered in the next chapter.

In this case, you will use the model created by the ModusToolbox™ ML Configurator, but you will still need the code to perform any preprocessing that was done to the data in Imagimob, such as a sliding window. You can get that by simply selecting **Preprocessor** from the **Code Gen** tab.



Architecture: Any (ANSI C99)
Output Name:
Output Directory: Gen
Timestamps API: Keep track of timestamps. Extend API with time_in and time_out parameters.
C Prefix: IMAI_IMU_
Build: Preprocessor

Generate data python bindings for C-library.
This will add a Makefile to build native .dll library and a python wrapper to output directory. For distribution only the .dll file and .py file is required.

Generate data compare test (C) for model.
This will add a Makefile and a data compare test runner (runner.c) to output directory.

Input file to use for compare test
File type: Imagimob Data file (.data)
File path:
Expected output data file to use for compare test
File type: Imagimob Data file (.data)
File path:

When you want to test the optimized model on the device, you can use the Machine Learning Imagimob MTBML Deploy (*mtb-example-ml-imagimob-mtbml-deploy*) code example. The difference is that you use the C models that the ML configurator creates from the Keras *.h5* model instead of the C files that came directly from Imagimob Studio. The C files created from Imagimob are still used, but only for preprocessing of the data before sending it to the inference engine.

The code example assumes that the **C prefix** for the models is **IMAI_IMU_** (shown above) for motion detection applications and **IMAI_PDM_** for audio applications.

In the code example, the same Imagimob API functions are used (with the new prefix) but they only perform preprocessing (e.g. `IMAI_IMU_init`, `IMAI_IMU_enqueue`, `IMAI_IMU_dequeue`). Once preprocessing is done, the ModusToobox™ API is used to run the inference engine (e.g. `mtb_ml_model_run`) on the preprocessed data.

To use the Keras *.h5* model and the *.c/.h* preprocessor files from Imagimob, copy them into the MTBML deploy application's *Edge/IMU* or *Edge/PDM* directory.

The code example supports hardware-specific optimizations including model quantization, sparsity support, and optimization for NPUs including NN-Lite or U55e.

3.2.9 Deploy in the application

Once you have the final model, you will want to deploy it in your end application. You can either add your application's functionality to the code example that was previously used to verify the model or you can integrate the model into another application. Either way, it is helpful to review how the model is used in the code example.

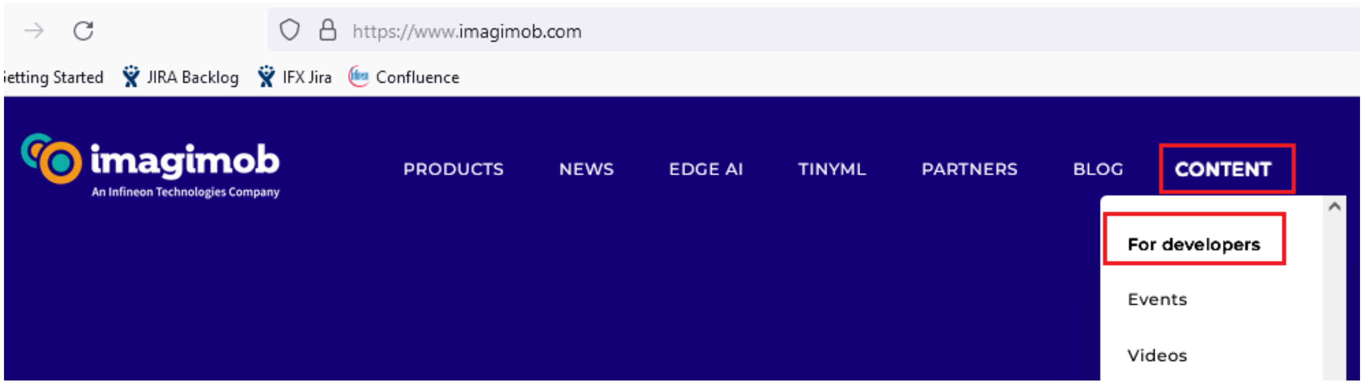
You can find documentation on the Imagimob API used to interface with the model from the website: https://developer.imagimob.com/edge-optimization/edge-api.

3.3 Getting Started

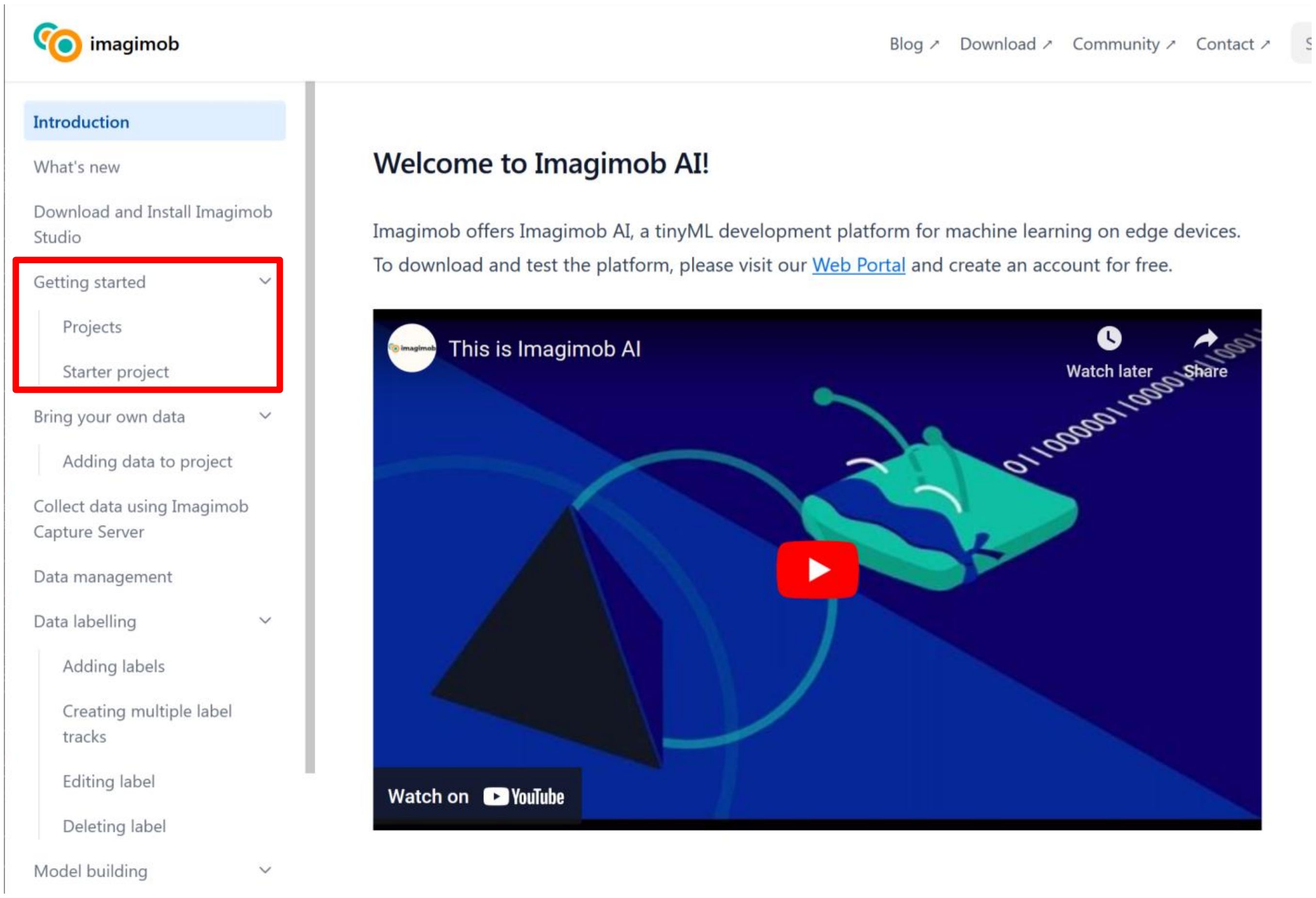
The Imagimob website contains lots of useful documentation. Rather than repeating that information here, let's take advantage of their existing documentation. First, go to the Imagimob main webpage:

https://www.imagimob.com/

From that page, select **Content** and then select **For Developers** from the drop-down menu.



The developer page includes links for Getting Started, Data management, Data labelling, Model building, and Edge optimization.

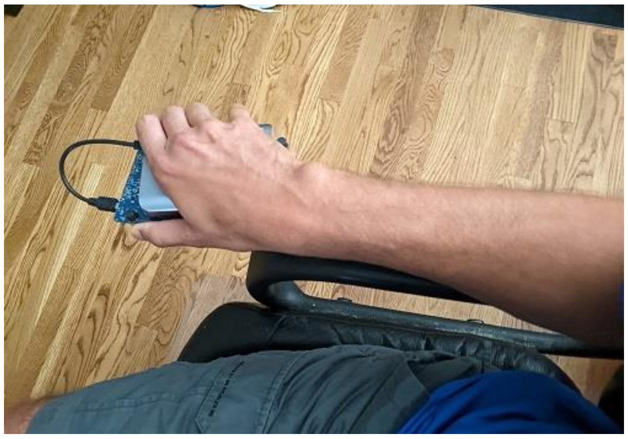


Welcome to Imagimob AI!

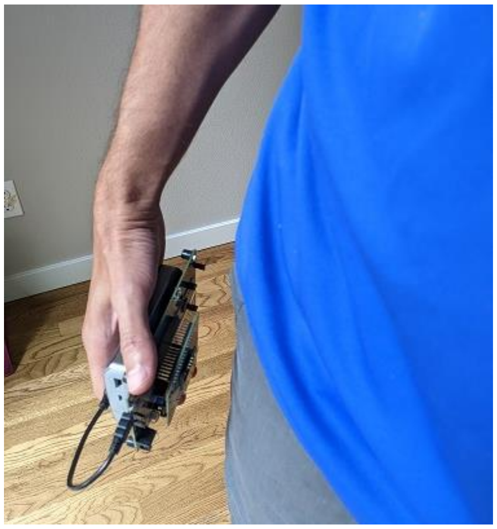
Imagimob offers Imagimob AI, a tinyML development platform for machine learning on edge devices. To download and test the platform, please visit our Web Portal and create an account for free.

The documentation provides a tutorial that goes through the entire Imagimob flow for one of the starter projects. The Human activity recognition project provides a good introduction as it uses the motion sensor to detect movement of the kit. It will detect the following movement conditions: sitting, standing, walking, running, and jumping.

The orientation of the kit is important for proper motion detection. The existing data was collected based on the orientations described below. For sitting and standing the positions are shown in the images below. For walking, running, and jumping the board is held the same way as standing with the normal arm movements associated with each respective activity.



* Sitting: Kitprog USB facing forward, shield toward the ground



* Standing: Kitprog USB facing forward, shield toward the body

3.3.1 Account types

If you don't already have one, you will need to create a free Imagimob evaluation account in the first exercise. This account never expires and supports many of Imagimob's tools and features, limited to 300 cloud compute units on Imagimobs's servers. Don't worry – that's plenty of time to complete the exercises in this chapter. For additional account options, visit: https://www.imagimob.com/contact-sales.

3.4 Exercises

Exercise 1: Install Imagimob tools

Note: If you already did the installation in Chapter 1 you can skip this exercise.

☐ 1. Create an Imagimob account: https://account.imagimob.com/signup.

☐ 2. Download and install Imagimob Studio.

Note: Imagimob Studio is only available for Windows.

☐ 3. Clone the Imagimob Capture Server https://bitbucket.org/imagimob/captureserver.

☐ 4. Install Python if it is not already installed on your system. It is recommended that you use Python version 3.11 or later.

Note: It is recommended to go to https://www.python.org/downloads/ to download the latest version.

☐ 5. Go to the *captureserver* repo directory and run `python setup.py install`.

Note: If you are operating on Windows, you can use either modus-shell or a Windows command prompt.

☐ 6. Download and install the Imagimob Capture App on an Android phone.

Note: The Imagimob Capture App is only available for Android devices.

Exercise 2: Use the complete Imagimob flow

In this exercise, you will try out the full Imagimob flow using the Imagimob Human Activity Recognition starter project.

The various starter projects are described at https://developer.imagimob.com/getting-started/starter-project. The Human Activity Recognition starter project already contains a set of data, labels, and models, but you will go through each of the steps to collect additional data, add it to the Imagimob project, label it, retrain the models, and test the best model on the device.

Note: *Detailed documentation for each step can be found along the left side of the https://developer.imagimob.com/ site. You can also refer to the Workflow section in this document.*

Collect data

☐ 1. Create the Machine Learning Imagimob Data Collection (mtb-example-ml-imagimob-data-collection) application in ModusToolbox™. Name the application **ch05_ex02_imagimob_collection**.

☐ 2. In the file *Makefile*, verify that the `DEFINES` variable has the correct shield selected.

Note: *If the silk screen on the back of the shield says "9-axis IMU" and "600-60608-01 REV03" or earlier, it is SENSE_SHIELD. If the silk screen says "6-axis IMU" and "600-60608-01 REV04" or later, it is SENSE_SHIELD_v2.*

☐ 3. Program the application to your kit. This will allow the kit to be used to collect accelerometer data to use when adding additional data to the Imagimob projects.

☐ 4. Use the ModusToolbox™ data collection code example with the Python `captureserver` to collect data for sitting and standing.

Note: *See the code example's README.md for instructions on how to use the data capture server to collect data.*

Note: *If you have more than one camera, you can use `--opencv-camera-index` with a value other than 0. The second camera will typically have an index of 1.*

Import data, set up models, train and export

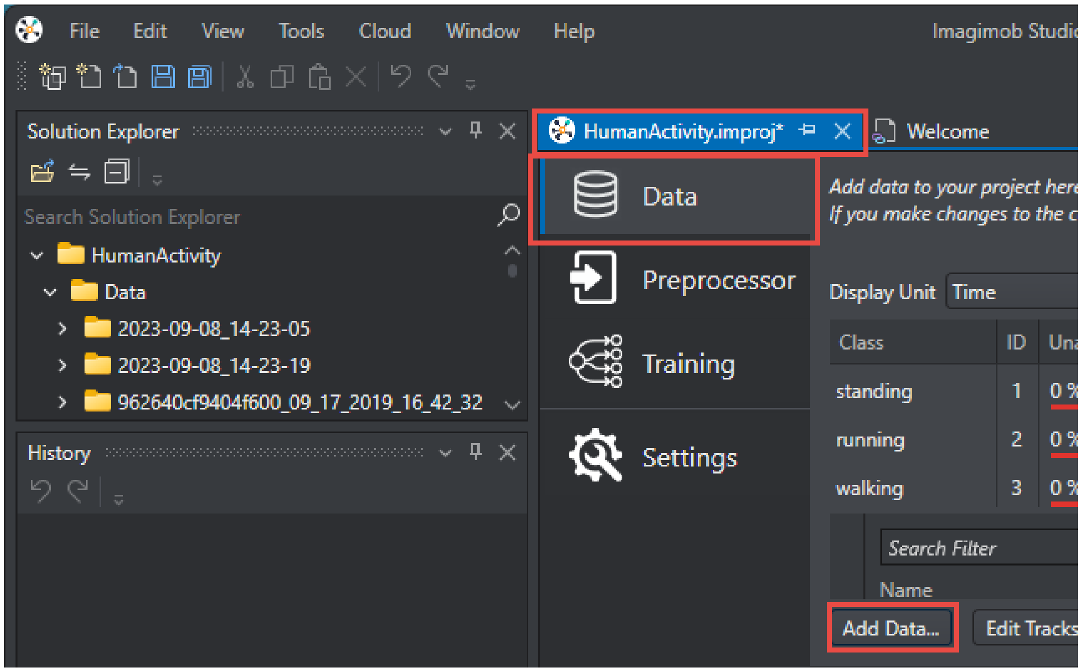
☐ 5. Create a new Classification project in Imagimob Studio based on the **Human Activity Recognition Infineon** project.

Note: *Details can be found at: https://developer.imagimob.com/getting-started/workspace-project.*

☐ 6. Copy the data you collected in the previous section into the Imagimob project directory structure.

Note: *The default directory for the Imagimob data is <Workspace>\HumanActivityInfineon\Data*

☐ 7. Open the project file (**.improj*) and select the **Data** tab. Add the new data to the project by using the **Add Data** button at the bottom of the **Data** tab window.



☐ 8. Click the **Add** button from the **Select import mode** page and the select the **Data** folder. Select **Nested structure** and click **Next**.

☐ 9. Click **OK** in the Batch Import window. You will see a message that indicates there are some segments with the same name already in the database. This occurs because you previously told it to search the entire Data directory so any sessions that were already in the project are found again. Click **Don't Replace** to only add the new sessions.

☐ 10. When the import is done, open the *.imsession* file for each new data session and add the appropriate labels.

Note: *Hold the left mouse button down and drag to select an area to add a label. Then right click in the label track and select **New Label.** The starting time and duration will be set to the region you selected.*

Note: *Once a label is created, you can drag it around or drag its endpoints to change the duration.*

Note: *You can scroll by holding the right mouse button and dragging. The bar at the bottom of the screen shows which section of the file is currently shown.*

Note: *The play head is always centered in the window so if you want to play back from a different location, just scroll the screen.*

☐ 11. Redistribute the data by clicking on **Redistribute Sets** at the bottom of the **Data** tab window.

☐ 12. Set up data pre-processing on the **Preprocessor** tab.

Note: *In this case, just review the pre-processing that has already been configured.*

☐ 13. Generate models from the **Training** tab by clicking **Generate Model List** at the bottom of the window.

Note: *Leave the values as-is but look at the different tabs (Auto ML, Training, Build Steps) to see the possibilities.*

☐ 14. Train the models by clicking **Start New Training Job** at the bottom of the **Training** tab window.

☐ 15. Open the job when prompted to view training progress and results.

Note: *You can also use **Cloud > Connect to Imagimob Cloud or** the **Open Cloud** button at the top right corner of the window and then double-click the job name to see the progress.*

Note: *When you connect to the cloud, there is a tab to see how many credits you have remaining in your account.*

☐ 16. Download the best model back into Imagimob Studio by clicking the **Download** button for the model you want to use.

☐ 17. Evaluate the model in Imagimob Studio.

Note: *Double-click the .h5 file in the Models/<name>/Predictions directory and select the **Evaluation** tab.*

☐ 18. Generate the *.c/.h* model files for use on the device.

Note: *Switch to the **Code Gen** tab. Review the selections and the click **Generate Code**.*

Note: *On Windows, the default location for the downloaded edge model is <workspace>/<project name>/Models/<model name>/Gen.*

Test on the device

☐ 19. Create the Machine Learning Imagimob Deploy (mtb-example-ml-imagimob-deploy) example in ModusToolbox™. Name the application **ch05_ex02_imagimob_model**.

☐ 20. Replace the *model.c* and *model.h* files in the application's *source* directory with the files that you downloaded.

☐ 21. Program the application to your kit and ensure the model is running properly.

Note: *Open a UART terminal with a baud rate of 115200 to see the model's predictions as you move the kit to mimic various motions.*

Exercise 3: Run some application examples

In this exercise, you will try out some other application examples provided by Imagimob.

The various applications showcase different features of Imagimob such as model creation using full AutoML (which you used in exercise 2), the advanced tuning UI for more control, and the Imagimob Capture App/Server for complete control over the data.

☐ 1. Review the list of starter projects via Imagimob Studio. Try out one or more examples to see how different features of Imagimob are used. A few options that may interest you are:

- The **Keyword Spotter** project that recognizes keywords with a microphone and MCU/CPU.
- The **Fall Detection** project that detects a fall with an IMU.
- The **Acoustic Recognition** project that uses a microphone to sample and identify specific acoustic events.

Published by

Infineon Technologies AG

81726 Munich, Germany