CoolThink@JC - Computational Thinking Education Programme

Course Code & Title

CF3E Computational Thinking Development - Level 3

Unit Code & Topic

L3UAI2

PICaboo

香港教育大學
The Education University of Hong Kong

Massachusetts Institute of Technology

Funded by

香港賽馬會慈善信託基金
The Hong Kong Jockey Club Charities Trust
同心同步同進 RIDING HIGH TOGETHER

These teaching and learning material were co-created by The Education University of Hong Kong, and Massachusetts Institute of Technology, and funded by The Hong Kong Jockey Club Charities Trust.

Table of Contents

1. Synopsis

Students will learn about the basics of machine learning and create their own apps that implement these concepts through image classification. Students will train an image classification model using a web-based interface. Students will then create a new mobile app using App Inventor. This app will emulate the game of Peekaboo that one plays with very young babies who are learning to make sense of the visual world around them. Students will use their trained model as the classifier within the app. Students will use MIT App Inventor's machine learning extension called the PersonalImageClassifier when creating this app.

Peekaboo: Very young babies do not quite understand yet how to make sense of the visual world around them. One of their challenges is the lack of the sense that objects continue their existence even when they are invisible. This concept is described by the famous child psychologist Jean Piaget as "Object Permanence". A game that parents often play with their very young babies in the US is called Peekaboo where a parent hides her face (with her hands or with a pillow etc.) and "disappear" to the surprise of the baby and then suddenly show her face saying "Peekaboo, I see you" much to the delight of the baby, who is astonished to see the parent suddenly appear out of "nowhere". Here is a video you can watch for this game (https://www.youtube.com/watch?v=Z37Ru-GRtks). The process through which a machine learns to make sense of visual data is not unlike the way babies learn visual cues. In both cases there needs to be many examples (large sets of training data) and the categories of examples need to be easily distinguishable from one another (visually distinct data classes).

2.Unit Intended Learning Outcomes (UILOs)

After completing this unit, students will be able to:

$UILO_1$: explain the basic concept of machine learning;

$UILO_2$: apply the computational thinking concepts of events, conditionals, and data structures in coding of the PICaboo app;

$UILO_3$: apply computational thinking practices of testing and debugging and algorithmic thinking in coding of the PICaboo app;

$UILO_4$: train, test, and revise an image classification model;

$UILO_5$: use the PersonalImageClassifier extension and the trained model in an image classification app using App Inventor.

$UILO_6$: explain the five Big Ideas of Artificial Intelligence.

3. Mapping with the Computational Thinking Framework

The following tables show the alignment of this unit with the intended learning outcomes of the computational thinking framework. The entries indicate the expected relevance of this unit to each outcome:

✓✓✓ : High relevance
✓✓ : Some relevance
✓ : Low relevance

Computational Thinking Concepts

L3UAI2: PICaboo		
1. Sequences	✓✓	The sequence of initialization of PersonalImageClassifier and retrieving model data is important.
2. Events	✓✓✓	GotClassification event for PersonalImageClassifier is the main event block for the app.
3. Repetition	✓	The PersonalImageClassifier once started will do continuous image classification from the video stream it receives.
4. Conditionals	✓✓✓	Conditionals are used to compare the PIC results of the user's facial postures and decide which image of the baby (happy or sad) to display.
5. Parallelism		
6. Naming and Variables	✓✓	Local variables are named appropriately to identify the best classification and store confidence levels of each of the classes.
7. Operators	✓✓	Bar graphs of confidence levels are computed using multiplication.
8. Data Structures	✓✓✓	The classification result is a dictionary data structure storing key-value pairs.

Computational Thinking Practices

L3UAI2: PICaboo		
1. Reusing and remixing	✓✓	The template is partially completed for students.
2. Being incremental and iterative	✓✓	Students iterate when training the model.
3. Abstracting and modularizing	✓	The PIC classification interface hides a lot of the complexity behind blocks that are easy to use.
4. Testing and debugging	✓✓✓	Students test their model on the Personal Image Classifier website, and also test their model in the app.
5. Algorithmic thinking	✓✓✓	Students complete an algorithm to graphically display confidence levels and make decisions about the machine's response to these confidence levels.

Computational Thinking Perspectives

L3UAI2: PICaboo		
1. Expressing	✓✓	Students can express themselves with their choice of facial postures while providing images for the training of the model.
2. Connecting	✓✓✓	Students connect image classification to technology they use daily.
3. Questioning	✓✓✓	Students question how machine learning works and how it can be developed and improved upon.
4. Computational identity	✓✓	Students build their computational identity by training their own ML model.
5. Digital empowerment	✓✓✓	Students are empowered to learn more about AI.

AI Five Big Ideas

L3UAI2: PICaboo		
1. Perception	✓✓✓	The app classifies images.
2. Representation & Reasoning	✓✓✓	Images are represented by numbers in a neural network to train the model.
3. Learning	✓✓✓	The model learns with the addition of more and better images.
4. Natural Interaction	✓✓	The app captures pictures of the user to classify and uses confidence levels to decide how to respond to the human user.
5. Societal Impact	✓✓	Algorithmic bias and potential for errors in machine learning models are discussed.

4. Learning Prerequisites

Students should have completed Levels 1-2 of the CoolThink@JC Computational Thinking Education Programme curriculum units, as well as the unit of L2UAI1 Voice Calculator, and have a basic understanding of computational thinking through the use of Scratch and App Inventor.

5. Hardware Prerequisites and Setup

Important Note: Though the students will not be creating apps until Lesson 2, it is important to test whether the apps will work at all beforehand. The extension requires the machine learning code to execute directly on the mobile device, and some older devices cannot handle it due to hardware limitations. Follow the instructions below under Lesson 2 to test whether the Personal Image Classifier and PICaboo app will work on the classroom mobile devices. Here is a list (http://appinventor.mit.edu/explore/ai-compatible-devices) of device/OS pairs that are known to work with this extension.

Lesson 1

For Lesson 1, each student will need individual access to a computer that is connected to the internet. They will need to be able to open and interact with Personal Image Classifier (https://classifier.appinventor.mit.edu/) in a browser. Note that Lesson 1 needs neither App Inventor nor mobile devices.

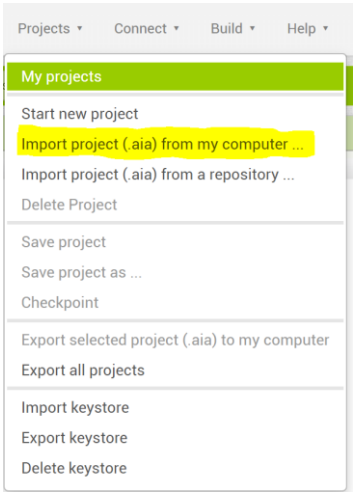
Lesson 2

For Lesson 2, each student will need individual access to a computer or laptop that is connected to the internet. They will also need Android mobile devices (smartphones or tablets) that have cameras and can run App Inventor's Personal Image Classifier Extension (noted in previous section). Chromebooks may also be a help to run both App Inventor and the AI Companion in lieu of mobile devices. Below is a list of device/OS pairs (http://appinventor.mit.edu/explore/ai-compatible-devices) that are known to work with this extension.

Make sure that the mobile devices have the latest updates and have the MIT AI2 Companion installed so that students can run the app on the device (https://play.google.com/store/apps/details?id=edu.mit.appinventor.aicompanion3).

If you want to run the PICaboo app later in the device which is not on the above compatibility list, you should check beforehand if the PIC extension will work on your device. (Older devices might not be able to run the extension.) You can download this LookTest.apk file (http://appinventor.mit.edu/assets/files/LookTest.apk) and install it on your device(s). Skip to the Test App instructions below to see how to test the app.

You can also test using the completed project file and the AI Companion. Start by downloading the completed project file (PICaboo_Completed.aia) onto your computer. Then, open App Inventor and import the aia file you just downloaded by clicking on "Projects" and then "Import project (.aia) from my computer" as shown below.



To connect your app to a mobile device, keep the imported project open and click on "Connect" and then "AI Companion" as shown below.

Open the MIT AI2 Companion on the mobile device and connect either with a code or by scanning. The app should show up on the mobile device.

Test App

Click the Start button and toggle the camera view as needed. You should see a video feed appear in the app. You should also see the classification confidence levels appear as bar graphs at the top of the screen. If video feed does not appear, it may mean that the device's hardware is not sufficient to run the extension. If that is the case, you may not be able to run Lesson 2 with students, because you will not be able to run the apps in class.

When students are ready to start Lesson 2, they will use this template aia file (PICaboo_Template.aia) which they will work on.

6. Pedagogical Approaches

The suggested pedagogical approaches of this unit are as follows:

- Class discussion
- Class demonstration
- Worksheets
- Guided individual coding
- Testing and debugging

7. Lesson Plan

This unit consists of 2 lessons of 70 minutes (two 35-minute parts for each lesson), with post-lesson activities after each period.

Lesson 1

Time	Activities	Learning Materials
5 mins	**To Play: Introduce PICaboo app** 1. Play video "peekaboo" to explain how a baby plays with parents.	Video "peekaboo" (https://www.youtube.com/watch?v=Z37Ru-GRtks)
15 mins	**To Play: Try PICaboo app** 1. Let students try PICaboo app with the help of *Worksheet 1: Discovery Worksheet* to understand the app's capabilities.	-PICaboo_Completed.apk file - *Worksheet 1: Discovery Worksheet*
10 mins	**To Think: Introduce Machine Learning** 1. Ask students "As the computer just runs what a human tells it, how can you train the computer to classify images by itself?" for discussion.	
15 mins	**Class Activity: Worksheet 2** 1. Let students complete the *Worksheet 2: Classification* to understand how a computer learns from the data. 2. Explain to students that a computer, like the concept of worksheet, can learn from the existing data to distinguish new data	*Worksheet 2: Classification*
20 mins	**To Code: Personal Image Classifier** 1. Let students try the web-based Personal Image Classifier with the help of *Student Guide: Lesson 1*: (1) create classes and labels; (2) capture images with webcam for sampling; (3) train (generate) a model;	*Student Guide: Lesson 1* -Pages 1-2 -Pages 3-4 -Pages 5-7

	(4) Figure out the poor performance of model for revising 2. Ask students to click "Export Model" and "Export Training Data" buttons to export the model.	-Page 8 -Page 9
5 mins	**Wrap-up** 1. Conclude that how a computer, like the concept of worksheet, can learn from the existing data to distinguish new data. 2. Notice that the PICaboo app will be built in the next lesson.	
Post-Lesson Activities	1. Complete the *Worksheet 2: Classification* if not completed in class. 2. Complete the training of the model if not completed in class.	*Worksheet 2: Classification*

Lesson 2

Time	Activity	Learning Materials
5 mins	**Review of Image Classification** 1. Check the answers of *Worksheet 2: Classification*, and explain the concept of image classification. 2. Explain to students that such model is going to be applied to the app via MIT App Inventor.	*Worksheet 2: Classification*
10 mins	**To Think: Introduce PICaboo app** 1. Explain the workflow of PICaboo app with the help of the *Student Guide: Lesson 2*. 2. Display the Designer and Blocks of PICaboo_Template.aia file. 3. Explain the built components of layout and code block sets.	*Student Guide: Lesson 2* -Page 4 -Page 2 -PICaboo_Template.aia file -Pages 3, 5-6
20 mins	**To Code: Build PICaboo app** Ask students to: 1. Open the PICaboo_Template.aia file.	

	2. Upload the model file (.mdl) exported in Lesson 1 to the component of Personal Image Classifier in Designer. 3. Build the code block set of when PersonImageClassifier1.GotClassification with the help of *Student Guide: Lesson 2*: (1) initialize local variables; (2) BarGraphs and Percentages; (3) if-the-else conditional statement. 4. Export, install, and test the PICaboo app in the device for debugging.	-PICaboo_Template.aia file -PICaboo_Completed.mdl file -PICaboo_Completed.zip file *Student Guide: Lesson 2* -Pages 10-11 -Pages 12-15 -Pages 16-18 -Pages 19-20
5 mins	**Review of PICaboo App** 1. Ask students to illustrate the PICaboo app completed in Lesson 2. 2. Ask students "What kind of concept else do you know about AI besides machine learning?"	
20 mins	**To Reflect: Five Big Ideas of AI** 1. Explain the five ideas of AI with the help of *Worksheet 3: Five Big Ideas of AI* and *Powerpoint: Image Classification*. 2. Discuss with students: (1) What else can the machine learning of AI help on our daily lives? (2) What kinds of problems may be raised by the wide-spread use of AI machine learning?	*-Worksheet 3: Five Big Ideas of AI* *-Powerpoint: Image Classification.*
10 mins	**To Reflect: Challenging Task** 1. Ask students, "What kinds of interesting apps else can you design based on the PICaboo app?" 2. Explain that the PICaboo app can be reused and remixed to another 3-class image classification app.	*Student Guide: Lesson 2* -Pages 21-22

Post-Lesson Activities	Encourage students to create another image classification app.	

8. Instructional Details

The teaching and learning materials of the three lessons are presented as below.

Lesson 1

1. Teacher's Guide: Lesson 1
2. Student Guide: Lesson 1
3. Worksheet 1: Discovery Worksheet
4. Worksheet 1: Discovery Worksheet (answer)
5. PICaboo_Completed.apk file

Lesson 2

1. Teacher's Guide: Lesson 2
2. Student Guide: Lesson 2
3. Worksheet 2: Classification
4. Worksheet 2: Classification (answer)
5. PICaboo_Template.aia file
6. PICaboo_Completed.aia file
7. PICaboo_Completed.mdl file (sample)*
8. PICaboo_Completed.zip file (sample)*
9. Worksheet 3: Five Big Ideas of AI
10. Worksheet 3: Five Big Ideas of AI (answer)
11. Powerpoint: Image Classification

**Teachers may need to prepare the sample files of trained model and data, generated from the Personal Image Classifier, by themselves for demonstration and for the students who forget to bring to lessons, or even lose or crash the files.*

9. Assessment

The following table maps the assessment modes and corresponding Unit Intended Learning Outcomes (UILOs) in Section 2.

Assessment Modes	UILO
Self-assessment: (1) Online multiple-choice questions (2) Survey of learning attitudes	$UILO_{3,4,5}$

Self-assessment

Self-assessment allows students to reflect on their learning performance. There are two sorts of self-assessment questions below:

Online Multiple-choice Questions

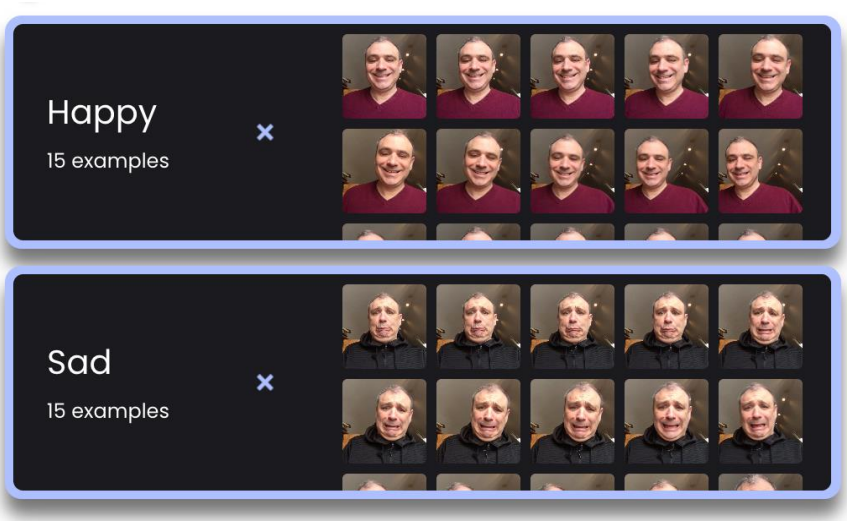
Online multiple-choice questions are adopted to assess students' understanding of the key concepts of the activity, as shown below.

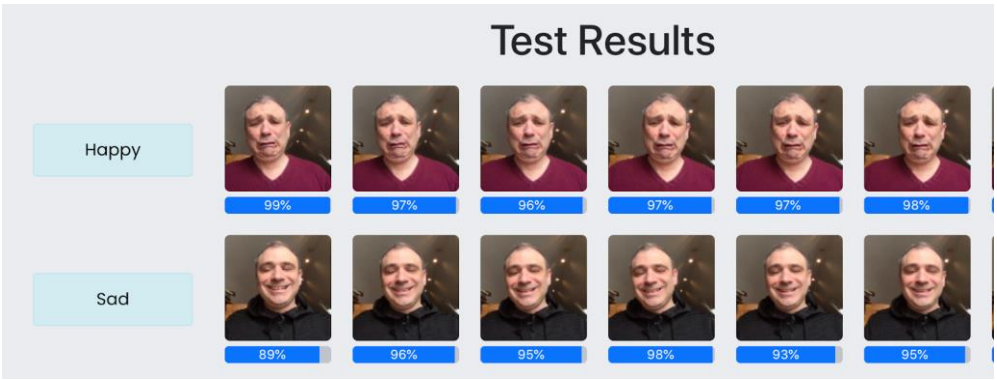
1. You train a model on facial postures of showing and hiding your face on the Personal Image Classifier (https://classifier.appinventor.mit.edu), and build the PICAboo app with your model. You play the game and each time the machines respond to your facial postures perfectly. You give the app to your friends, but the performance of the app is generally very bad. What is the best method to improve your app?

   A. Ask you friends using the app to wear clothes similar to yourself.
   B. Retrain the model with many people showing and hiding their faces using different clothes, backgrounds and lighting.
   C. Have your friends using the app use the same background you had during the training.
   D. Have your friends using the app use the same type of lighting you had during the training.

(Answer: B)

2. You train a model for 2 labels (Happy and Sad) using the Personal Image Classifier and get all incorrect labels during the testing phase. Looking at the results below, it classifies all the "Happy" and "Sad" faces incorrectly. What might be the problem?





A. Not enough test cases.

B. You need a third label in addition to Happy and Sad.

C. In the training phase all happy images included a red sweater; all sad images included a black sweater, so it incorrectly classifies a red sweater as happy even though the facial expression is sad.

D. There is no problem. You can expect a few incorrect labels during testing.

(Answer: C)

3. What would be the best way to make this model of Question 2 more reliable?

   A. Retrain the model with happy and sad photos where you do not wear different tops.
   B. Retrain the model with happy and sad photos where you wear different coloured tops for both labels.
   C. Only use the model on photos where you are wearing black.
   D. There is no problem. You can expect a few incorrect labels during testing.

(Answer: B)

4. Why is there a multiplication with 100 in this code block?

   A. Confidence values for a classification are returned as a decimal between 0 and 1 and they need to be converted to a percentage.
   B. Otherwise the bar graph would be too small to see.
   C. The width of the app screen cannot be larger than 100 pixels.
   D. It is an arbitrary choice.

(Answer: A)

5. Which of the following step is necessary to create a good image classification model?

   A. Adding lots of photos for training.
   B. Testing the model on new photos.
   C. Analyzing the test results to modify the training process.
   D. All of the above are necessary.

(Answer: D)

Survey of Learning Attitudes

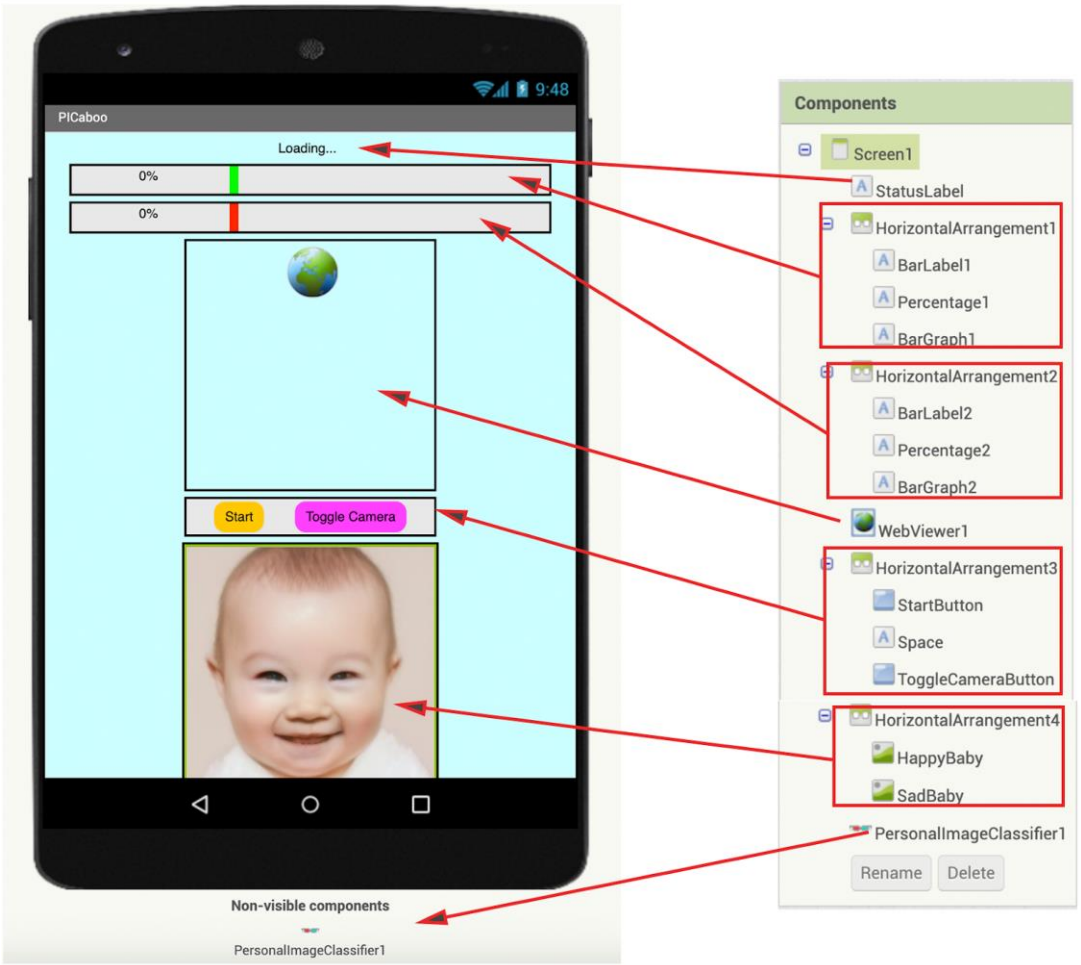
To evaluate students' attitudes, perceptions and understandings of coding, they are required to complete the following questionnaire by putting a "✓" in the appropriate boxes.

After completion of this unit, I think…	**Disagree**	**Somewhat disagree**	**Neutral**	**Somewhat agree**	**Agree**
Learning how to make apps makes me want to learn more about coding.					
I feel more connected to the technology around me when I make apps.					
I am excited to share this app with friends and family.					

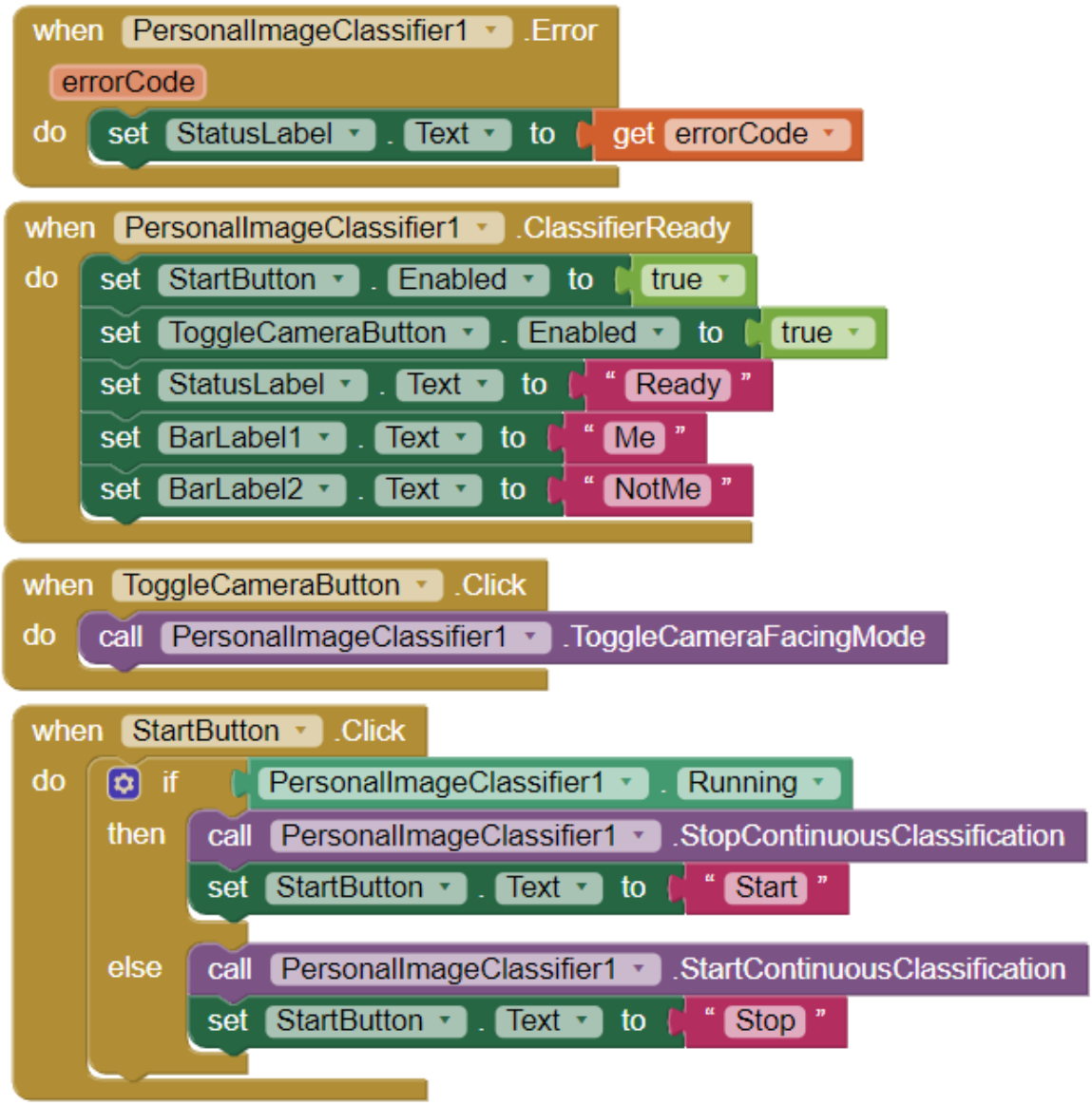
10. Screen Design and Code

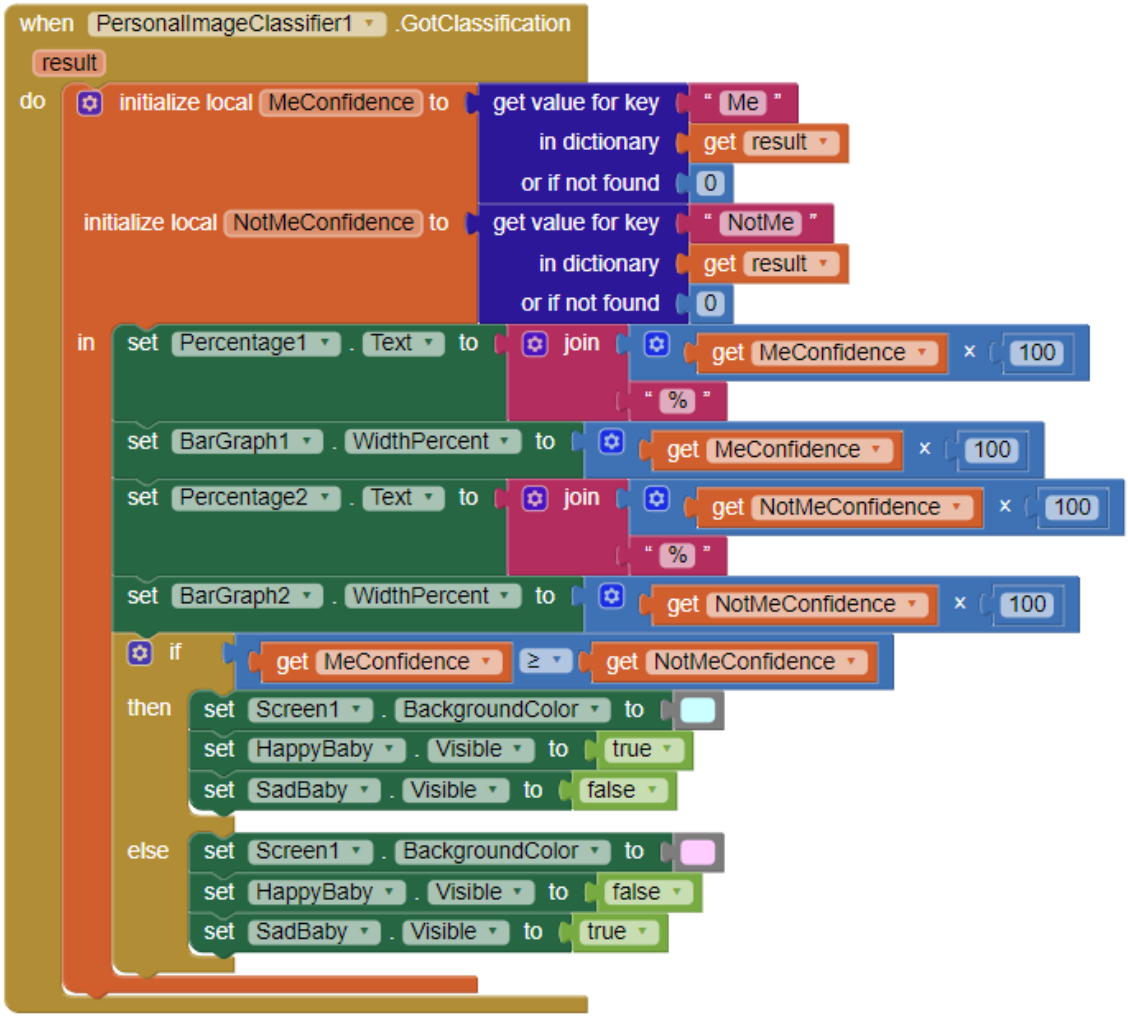
Designer

Screen1



Blocks


when PersonalImageClassifier1 .Error
errorCode
do set StatusLabel . Text to get errorCode
when PersonalImageClassifier1 .ClassifierReady
do set StartButton . Enabled to true
set ToggleCameraButton . Enabled to true
set StatusLabel . Text to " Ready "
set BarLabel1 . Text to " Me "
set BarLabel2 . Text to " NotMe "
when ToggleCameraButton .Click
do call PersonalImageClassifier1 .ToggleCameraFacingMode
when StartButton .Click
do if PersonalImageClassifier1 . Running
then call PersonalImageClassifier1 .StopContinuousClassification
set StartButton . Text to " Start "
else call PersonalImageClassifier1 .StartContinuousClassification
set StartButton . Text to " Stop "


when PersonalImageClassifier1 .GotClassification
result
do
initialize local MeConfidence to
get value for key " Me "
in dictionary get result
or if not found 0
initialize local NotMeConfidence to
get value for key " NotMe "
in dictionary get result
or if not found 0
in
set Percentage1 . Text to join
get MeConfidence × 100
" % "
set BarGraph1 . WidthPercent to
get MeConfidence × 100
set Percentage2 . Text to join
get NotMeConfidence × 100
" % "
set BarGraph2 . WidthPercent to
get NotMeConfidence × 100
if
get MeConfidence ≥ get NotMeConfidence
then
set Screen1 . BackgroundColor to
set HappyBaby . Visible to true
set SadBaby . Visible to false
else
set Screen1 . BackgroundColor to
set HappyBaby . Visible to false
set SadBaby . Visible to true

Appendix 1

L3UAI2 Teacher's Guide: Lesson 1

Teaching Objectives

1. Guide students the basic understanding of what machine learning is.
2. Guide students to gain the practical knowledge of training a model in image classification.

Lesson Outline

To Play: Introduce PICaboo app (5 minutes)

Play the video "peekaboo" (https://www.youtube.com/watch?v=Z37Ru-GRtks) to explain how a baby plays with parents: When the parents show the face in front of a baby, the baby will be surprised as he/she thinks the parents suddenly appear.

To Play: Introduce PICaboo app (15 minutes)

Let students try the PICaboo app with the help of *Worksheet 1: Discovery Worksheet* to understand the app's capabilities. Also, there is a YouTube video describing the practical use of AI image classification provided in the worksheet. Teachers can show it in the classroom for students to complete the questions. If time is not allowed, the questions can also be completed by class discussion, or even completed after school.

To Think: Introduce Machine Learning (10 minutes)

Ask students to discuss: "As we know that a computer just runs what a human tells it, how can you train the computer to recognize images by itself?"

Class Activity: *Worksheet 2: Classification* (15 minutes)

1. Let students complete the *Worksheet 2: Classification* to understand how a computer learns from the data.

2. Explain to students that a computer, like the concept of worksheet, can learn from the existing data to distinguish new data.

3. There will be another detail present on Machine Learning in the Lesson 2, while the worksheet just stimulates students to think.

To Code: Personal Image Classifier (20 minutes)

1. Ask students to try the web-based Personal Image Classifier (PIC) (https://classifier.appinventor.mit.edu/) to build a model of image classification with the help of Student Guide: Lesson 1:

   (1) Create: create the labels "Me" and "NotMe" (no space between "Not" and "Me") for two facial poses by clicking the cross symbol;

   (2) Add: click "Capture" to add about 25-50 images (more images increase accuracy but beware of lesson time) with webcam for each class to be the training data;

   (3) Train: Click "Train" button to generate a model according to the examples (Note that the Loss number is a measure of how well the model performs with the given images, and students should note how high or low that number is).

   (4) Figure out the poor performance of model for revising:

      i check the image examples of each class, and delete the less relevant images (ambiguous pose, poor lighting, fancy background, out of focus, and so on);

      ii add more images with satisfactory facial expressions to improve the model;

      iii Click "Train" button to generate the model again.

2. Ask students to click "Export Model" and "Export Training Data" buttons, when being satisfied with the performance of model, to build the files of model (.mdl) and training data (.zip) for saving the result, further analysing, and applying to the mobile app (note that the Personal Image Classifier cannot be auto-saved and all will disappear when closing the web browser).

Wrap-up

1. Conclude that how a computer, like the concept of worksheet, can learn from the existing data to distinguish new data.

2. Notice that the PICaboo app will be built in the next lesson.

Post-Lesson Activities

1. Complete the *Worksheet 2: Classification* if not completed in Lesson 1.
2. Complete the model of Personal Image Classifier if not completed in Lesson 1.

Appendix 2

L3UAI2 Teacher's Guide: Lesson 2

Teaching Objectives

1. Focus: Guide students to build a mobile app with the integration of a trained image classification model.
2. Guide students to code by using variables and conditionals.
3. Focus: Guide students to understand the main concept of five big ideas of AI.
4. Encourage students to innovate other mobile apps by reusing and remixing the PICaboo app.

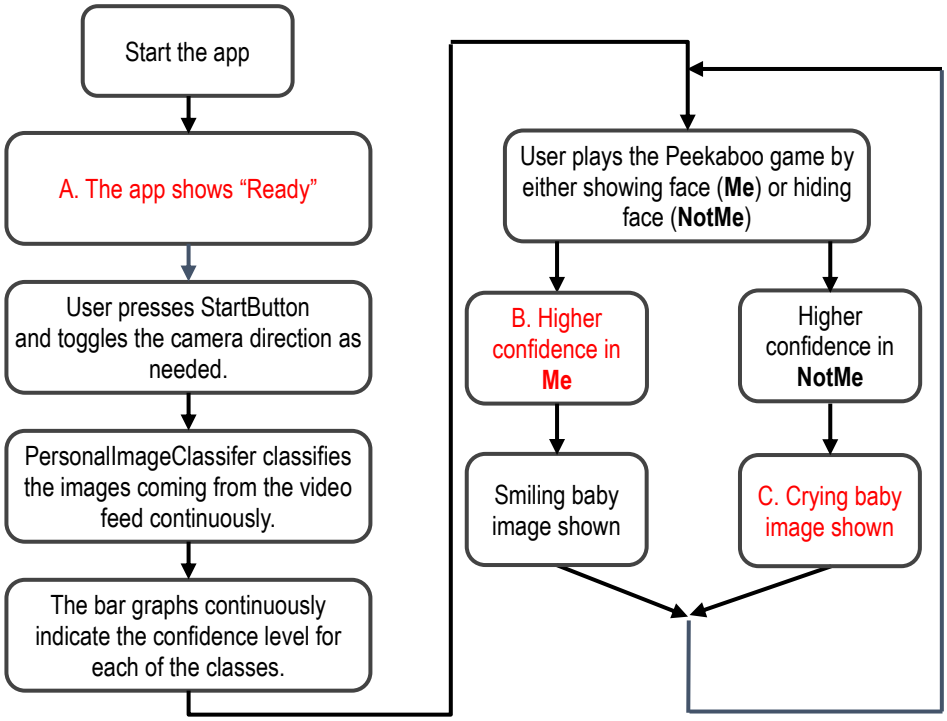
Lesson Outline

Review of Image Classification (5 min)

1. Check the answers of *Worksheet 2: Classification*, and explain the concept of image classification.
2. Explain to students that the model file (.mdl) built in Lesson 1 is going to be uploaded to the PICaboo app via MIT App Inventor for practical use.

To Think: Introduce PICaboo app (10 mins)

1. Explain the workflow of PICaboo app with the help of Page 4 of the *Student Guide: Lesson 2*.



2. Open the PICaboo_Template.aia file, and demonstrate the components of Designer and Blocks (Page 2).
3. Explain the functions of built components of layout and code block sets (Pages 3, 5-6).

To Code: Build PICaboo app (20 mins)

Ask students to:

1. Open the PICaboo_Template.aia file.
2. Upload the trained model file (.mdl) to the component of Personal Image Classifier in Designer (teacher should have a copy to share with the students who forget to bring back or even lose or crash the model file).
3. Build the code block set of when PersonImageClassifier1.GotClassification with the help of *Student Guide: Lesson 2*:
   (1) initialize local variables (Pages 10-11);
   (2) BarGraphs and Percentages (Pages 12-15);
   (3) if-the-else conditional statement (Pages 16-18).
4. Export, install, and test the PICaboo app in the device for debugging (Pages 19-20).

Review of PICaboo app (5 minutes)

1. Ask students to illustrate their own PICaboo app completed in Lesson 2, and summarize whether this app is successful or not, and how to further improve it. Comparison or competition of apps is possible if time allows.
2. Ask students "What other kind of concepts do you know about AI besides machine learning?"

To Reflect: Five Big Ideas of AI (20 minutes)

1. Explain the 5 Big Ideas of AI by asking students to complete the *Worksheet 3: Five Big Ideas of AI*.
2. Explain the *Powerpoint: Image Classification*.
3. Discuss with students:
   (1) How else can machine learning and AI help in our daily lives?
   (2) What kinds of problems may arise with the wide-spread use of AI and machine learning?

To Reflect: Challenging Task (10 minutes)

1. Ask students, "What other kinds of interesting apps can you design based on the PICaboo app?"
2. Explain and demonstrate how PICaboo app can be remixed and reused to produce a 3-class image classification app.

Post-Lesson Activities

Encourage students to create another image classification app.

Appendix 3

Machine Learning Resources

What is Machine Learning?

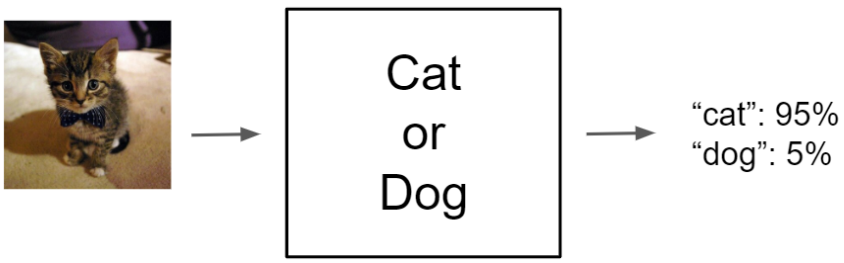
Machine Learning (ML) is a field of computation used to describe machines or systems that simulate human methods of cognition, such as learning or problem-solving. Examples of machine learning include Google's search engine, social media content suggestions, and the current development of self-driving cars.

How do machine learning and image classification work?

Image classification is a very important component of machine learning. For example, a self-driving car would have to instantly classify any images it sees, and the difference between a pedestrian and a green traffic light is critical.

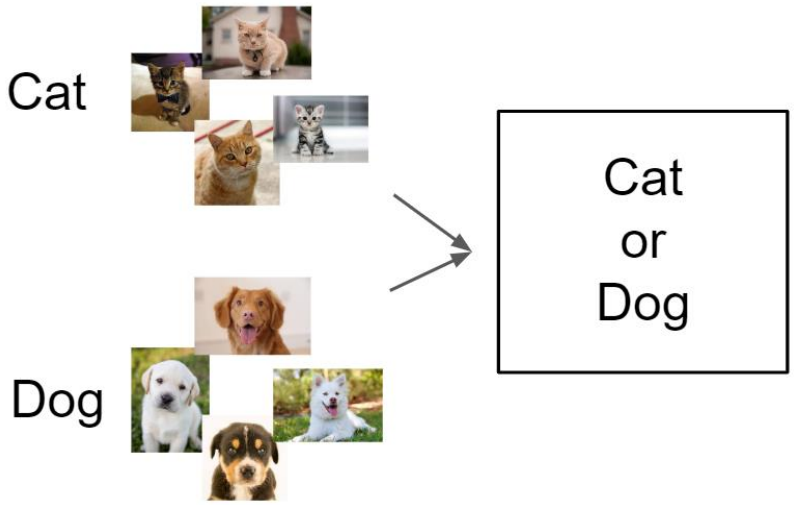
PersonalImageClassifier is an example of a **machine learning** tool that implements image classification. One popular way to create machine learning programs is by **training** them to classify images into some number of classes.

The overall goal for the program is to receive an image as an **input** and to give a label (a classification of the image) as an **output**. Let's say you want to classify images as either cats or dogs:



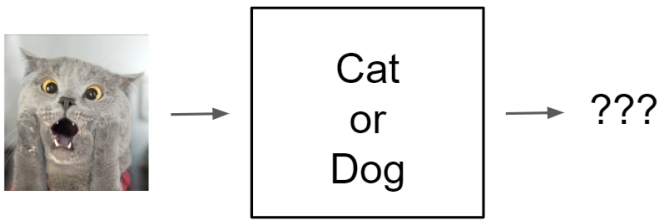
A lot of computation takes place in order to get from an image to the output text, and we will only describe the general principle here.

In order for the program to get started on this, you have to **train** it to classify certain images. For example, if you wanted to teach the program to tell whether an image is a cat or a dog, you would give it many images of both cats and dogs, labelled as such. It is important that you label each input image as a cat or a dog so that the program can know what kinds of images look like cats and what kinds of images look like dogs.

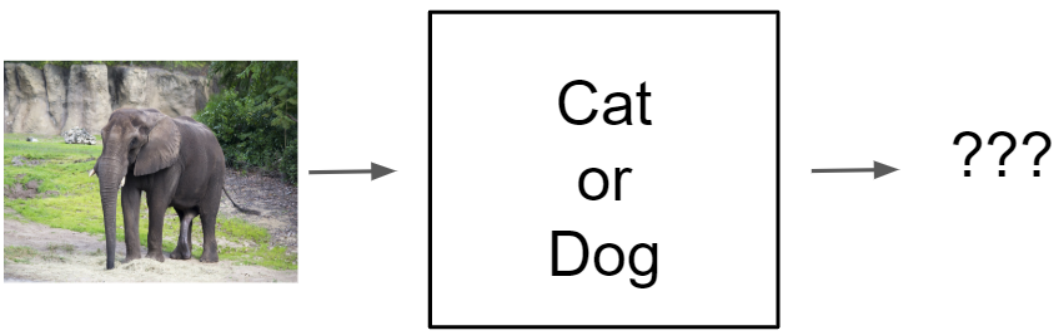


With enough examples, the program will be trained to "know" what types of images show dogs and what types of images show cats. It is important to note that this generally requires a huge number of examples from each class. Otherwise, the program will not have enough information to reliably classify new images.

Once the program has been trained with enough images, it can be **tested** by giving it a new image that we didn't use to train it and see what the program outputs.



Based on the results, you can go back to training, and continue the training and testing process for as long as needed.



Be aware that the program will only have the chance to correctly classify the images it was trained on. Since we trained with images labelled as “cat” and “dog”, any image that is tested will end up classified as a cat or a dog. Even putting in an entirely different image, such as an elephant, will be classified as a cat or a dog because those are the only two **classes** or **labels** of image it has been taught. A **class** (or **label** as it referred to in the Classifier web tool) is a possible classification or output. This means that the program will not be able to correctly classify an image if its class is not already in its database of trained classes.

Neural Networks

The PersonalImageClassifier uses a neural network to classify the input images. A neural network is effectively an interconnected series of nodes that all work together to create the output. A node is basically a series of computations, which could be random or specific computations.

When given an image, each node casts a “vote” of which of the classes the image is. This is then compared to what the image is actually labelled as. Correct nodes will be weighted more heavily in the future. For instance, given an image labelled as a “cat”, the nodes that correctly labelled it as a cat will be weighted more strongly in the future, while nodes that incorrectly labelled it as a dog will be weighted less since they now have a record of classifying incorrectly.

The idea is that with enough training, the adjusted program with the adjusted weights will do a good job in classifying not only the training images but also a lot of other images that it has not seen. Whether this idea actually works depends on many details, such as how the errors are computed, how the weights are adjusted, and how well the training images represent the entire set of images to be classified. All of these issues are active areas of study in machine learning research.

Supervised versus Unsupervised Learning

The PersonalImageClassifier uses **supervised learning**, which is a common way of classifying known categories. **Supervised learning** is when you have inputs (images) and you want to match them to known outputs (the labels “cat” and “dog”). This way, during training, it is easy to tell when an image is classified correctly or incorrectly.

Unsupervised learning is when you have inputs but no known outputs and you want to use machine learning to find patterns that are not already known. This could be used in a large collection of medical data if one is trying to find patterns that humans may have missed. Generally, image recognition programs will use supervised learning since we already know what images we are looking for.