

## Activity 1 – Hands on creating a QnA chatbot

In this activity, we will learn:

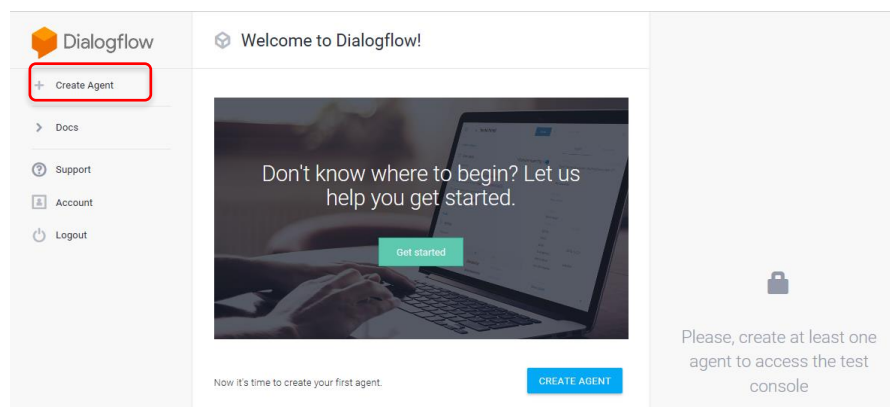
- ☐ Create DialogFlow Knowledge Base.
- ☐ Create an Integration to use a knowledge base
- ☐ Chat with the bot to verify the code is working
- ☐ Link Telegram to bot's channel

### 1. Prerequisites

- a) To start using Dialogflow Knowledge, you will need to have a Google account. Head over to here (<https://accounts.google.com/signup/v2>) to create one if necessary.

### 2) Create an Agent

- a) Login in to Dialogflow (<https://console.dialogflow.com/api-client/#/login> ).
- b) If it is the first time you login to Dialogflow with this account, you will be asked to allow Dialogflow to access your Google account. Click on **Allow** to continue.
- c) In the next screen, review your account settings. The only required action is to check “*Yes, I have read and accept the agreement*”. Click **ACCEPT** to continue.
- d) You may be asked to authenticate again. If everything is ok, you should see the following screen. Click on **Create Agent**

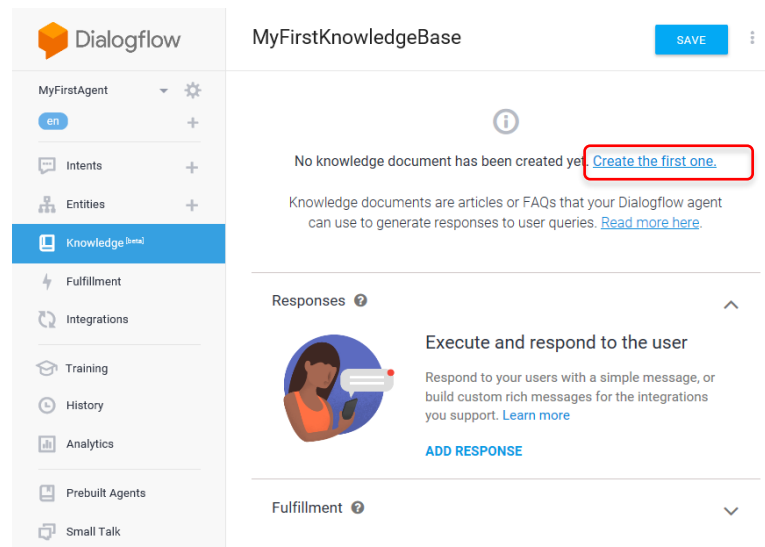


- e) Give your agent an appropriate name. We can leave the DEFAULT LANGUAGE and DEFAULT TIME ZONE settings. Click **CREATE** to continue. After a while, the screen will refresh and bring you to details configuration page.
- f) Click on **Setting** icon. In the **General** tab, turn on **BETA FEATURES** and click **Save**. Do note the **Dialogflow Knowledge** is still in beta at this point in time.

### 3) Create Knowledge Base

- a) On the left panel, click **Knowledge [beta]**, and then **CREATE KNOWLEDGE BASE** to create a new knowledge base.

- b) Enter a suitable knowledge base name and click on **SAVE**.
- c) Click on **Create the first one** to create a knowledge document. Knowledge documents are articles or FAQs that your Dialogflow agent can use to generate responses to user queries.



d) Enter values according to your data.

**Document Name:** this can be anything that you want

**Knowledge Type:** this can be selected as FAQ. For FAQ, only text/html and text/csv is supported for now.

- FAQ: The document content contains question and answer pairs as either HTML or CSV. Typical FAQ HTML formats are parsed accurately, but unusual formats may fail to be parsed. CSV must have questions in the first column and answers in the second, with no header. Because of this explicit format, they are always parsed accurately.
- Extractive QA: Documents for which unstructured text is extracted and used for question answering.

Can refer to <https://cloud.google.com/dialogflow/es/docs/how/knowledge-bases> for more details.

**MIME type:** this is the type of data you are going to feed to the bot. It has options like (text/plain, text/html, text/csv, application/pdf)

**Data Source:** this can be provided as a file on cloud storage or local, you can also give this as a URL to a public page. For our purpose of this activity, we will use a **URL**. Enter <https://www.merdeka.generation.sg/en/faqs>. You are free to try with another FAQ URLs. <https://token.gowhere.gov.sg/faq>.

Click **CREATE** to continue.

Create New Document

Document Name \*

My First FAQ

Knowledge Type \*

FAQ

Mime Type \*

text/html

DATA SOURCE

☐ File on Cloud Storage

gs://bucket-name/object-name

☒ URL

http://www.example.com/faq \*

https://www.merdeka.sg/en/faqs

☐ Upload file from your computer

SELECT FILE

☐ Enable Automatic Reload ?

CREATE

- e) Wait for it to generate all the data from the source that you have provided. This usually takes about 2–5 minutes depending upon the size of your data. You can see 'My First FAQ' knowledge base has been created.

Dialogflow

MyFirstKnowledgeBase

SAVE

Search documents

Document Name	Knowledge Type	Mime Type	Source/Path
My First FAQ ( View Detail )	FAQ	text/html	https://www.bicentennial.sg/frequently-asked-questions/

+ New Document

Responses ?

Execute and respond to the user

Respond to your users with a simple message, or build custom rich messages for the integrations you support. [Learn more](#)

ADD RESPONSE

Fulfillment ?

- f) You can look at the content of data inside FAQ by clicking on '**View Detail**'. It should look something like this

The screenshot displays the Dialogflow Essentials web interface. On the left is a sidebar menu with options: 0611Agent, Intents, Entities, Knowledge [beta] (selected), Fulfillment, Integrations, Training, Validation, History, Analytics, Prebuilt Agents, and Small Talk. The main area is titled 'My First FAQ' and contains a search bar, a table of question-answer entries, and a 'Docs' link at the bottom. The table has two columns: 'Question' and 'Status'. Two entries are visible, both with a status of 'ENABLED'.

Question	Status
<b>Question:</b> I am not eligible for means-tested MediShield Life premium subsidies because of high income, high AV, or multiple property ownership. Am I still eligible for the additional premium subsidies for the Merdeka Generation? <b>Answer:</b> All Merdeka Generation Seniors will receive additional MediShield Life premium subsidies of 5%, increasing to 10% after they turn 75 years old. These additional subsidies apply regardless of income, house type or property ownership.	ENABLED
<b>Question:</b> As a Merdeka Generation senior, should I still apply for CHAS? <b>Answer:</b> As a Merdeka Generation senior, you will already enjoy special Merdeka Generation CHAS benefits when visiting CHAS GP and dental clinics, with your MG card. However, there are two reasons why you might still wish to apply for a CHAS card: First, if you apply for CHAS, your household members can also receive CHAS cards. Second, when you apply for CHAS, your eligibility for higher healthcare subsidies will be assessed at the same time. Lower- to middle-income MG seniors – in other words, those who are eligible for Blue or Orange CHAS cards – will enjoy higher subsidies for subsidised services at the public Specialist Outpatient Clinics (SOCs), and subsidised outpatient medications. The additional 25% off that MG seniors receive will then be applied on	ENABLED

- g) Click the menu button (three vertical dots) beside the SAVE button and select **back** in the dropdown menu to return to the Knowledge base view.

#### 4) Create telegram bot

For most integration, you must provide configuration information to run your bot with Dialogflow. Most channels require that your bot have an account on the channel, and others, like Facebook Messenger, require your bot to have an application registered with the channel also.

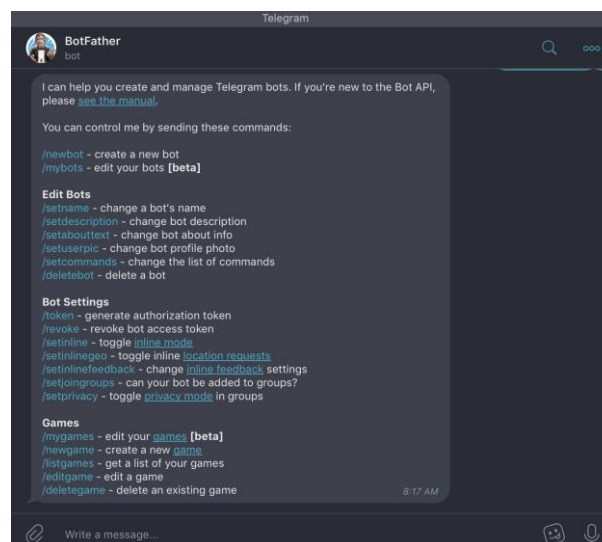
- If you do not have telegram installed on your mobile phone, please install via Google Play store or Apple App Store.
- If you have telegram installed on your laptop, use this [link](https://telegram.me/botfather) to connect to Bot Father (<https://telegram.me/botfather>) or add botfather.

Otherwise, if you are using telegram on your mobile, start telegram and search for BotFather.

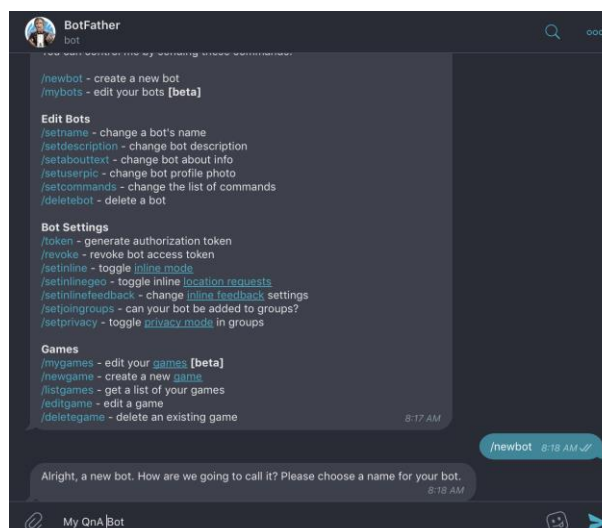
NOTE: There may be more than 1 BotFather. Select the one that indicate as **@BotFather, bot**



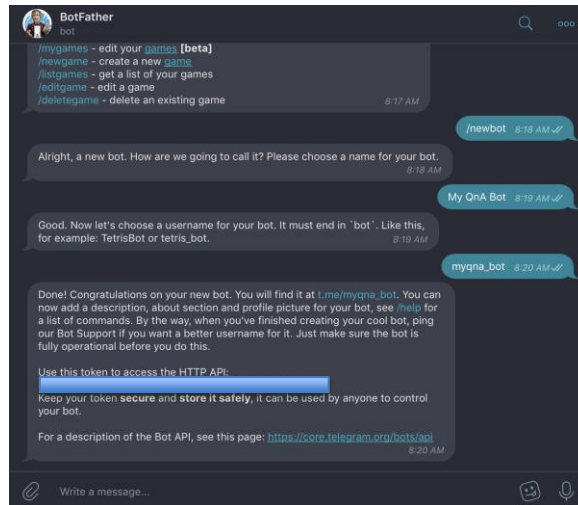
c) Click on **Start** to start a conversation with botfather.



d) Create a new Telegram bot by sending command /newbot.



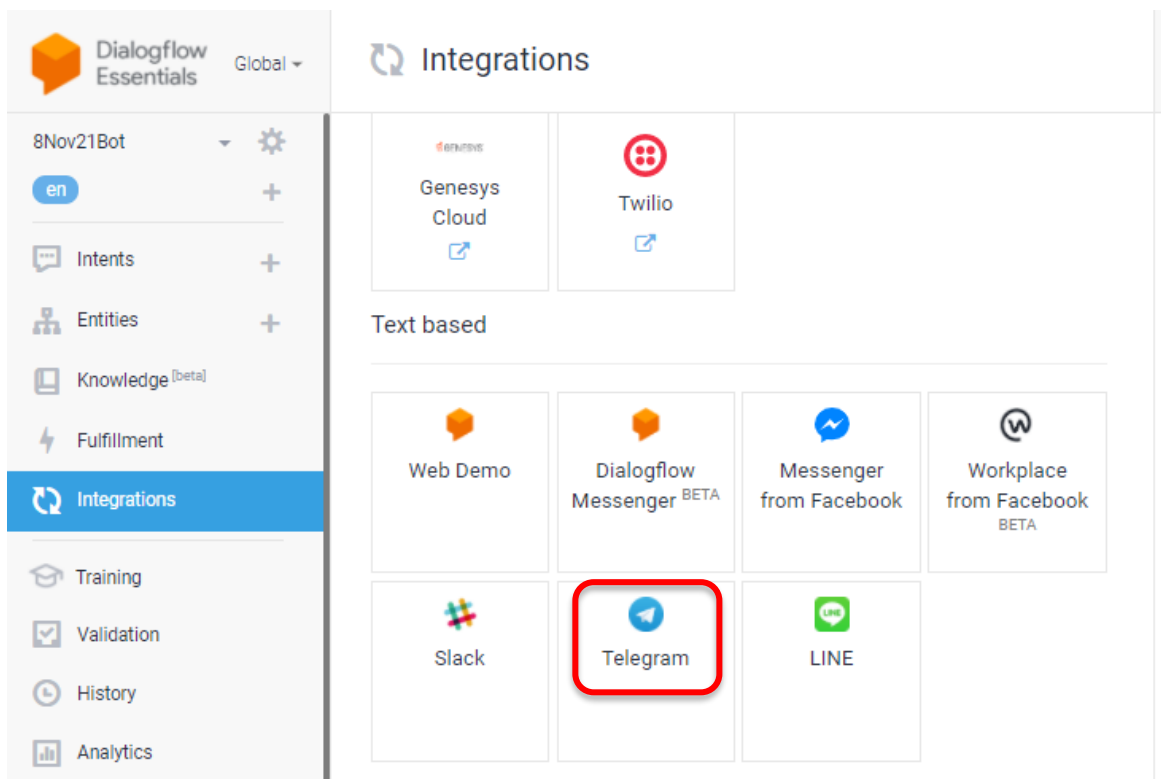
- e) Give the bot a friendly name.
- f) Specify a username. *\*Name cannot start with number and must end with `_bot`*



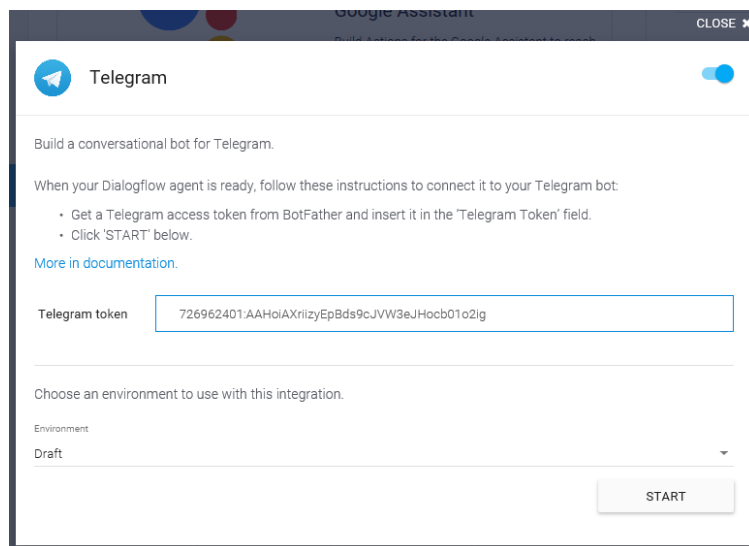
- g) Copy the Telegram bot's access token provide in the screen above. You will need this info to configure integration for the Agent.

## 5) Integrate telegram with Dialogflow

- a) Back to Dialogflow dashboard, click on **Integration**. click the icon **Telegram**.



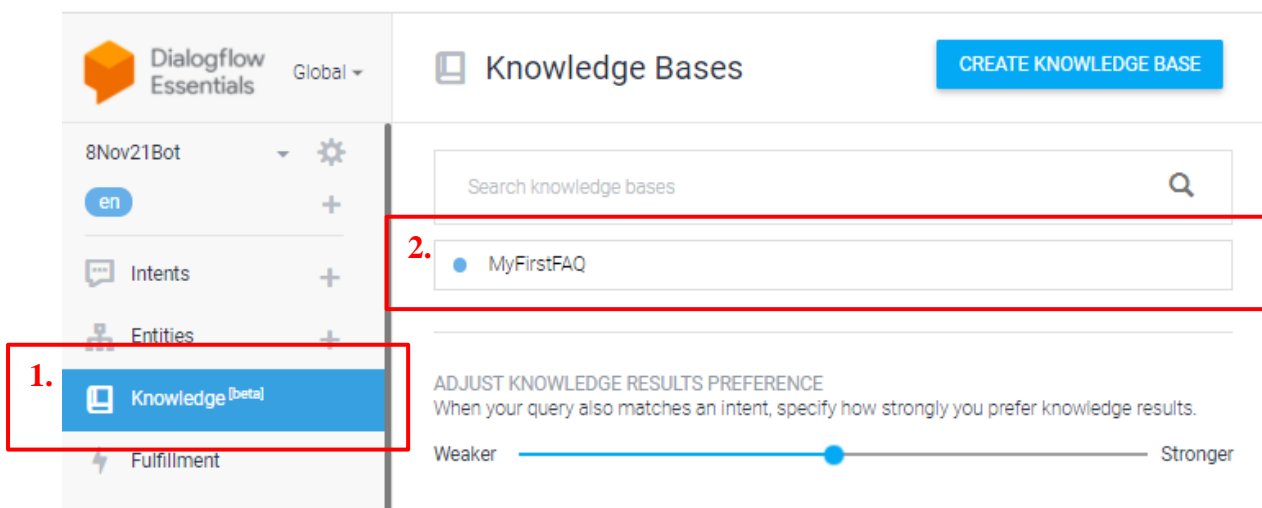
- b) In the Telegram configuration pop up, enter the telegram bot's access token you obtained in Step 4. Click **START** and after a while, you should see a flash message saying the bot is started.



c) Click on Close (top right) to finish the configuration.

## 6) Add Responses

a) From the menu, select Knowledge and click on the Knowledge Base you have created.



b) Click **Add Response** to create a new response.



**Dialogflow Essentials** US ▾


**MyFirstKnowledgeBase**

Search documents

Document Name	Knowledge Type	Mime Type	Source
MyFirstFAQ ( <a href="#">View Detail</a> )	FAQ	text/html	https://qs

[+ New Document](#)

**Responses** ?

 **Execute and respond to the user**

Respond to your users with a simple message, or build custom rich messages for the integrations you support. [Learn more](#)

[ADD RESPONSE](#)

- c) Click **+** and select Telegram to add to use responses from Telegram (if Telegram tab is not shown). Ensure the selection is turned on. Click **Save**.

**Responses** ?

DEFAULT **+**

Telegram


Google Assistant

1 \$Knowledge

2 Enter a

**Responses** ?

DEFAULT **TELEGRAM** × **+**

 Responses from this tab will be sent to the Telegram integration.

Use responses from the DEFAULT tab as the first responses.

☒

## 7) Small Talk

- a) Your agent can learn how to support small talk without any extra development. By default, it will respond with predefined phrases. (By default, if you send “How are you?”, agent will respond with “Sorry, can you say that again?”)

The screenshot shows the Dialogflow console interface. On the left is a sidebar with navigation options: MyFirstAgent, Intents, Entities, Knowledge (beta), Fulfillment, Integrations, Training, History, Analytics, Prebuilt Agents, and Small Talk (highlighted). The main area is titled 'Small Talk' and contains a 'SAVE' button in a red box. Below the title, there is a description of the Small Talk feature. A chat log shows a conversation: User: How are you? Agent: Wonderful as always. Thanks for asking. User: You're so sweet. Agent: Thanks! The feeling is mutual. Below the chat log is an 'Enable' toggle switch, which is turned on and highlighted with a red box. A warning message states: 'Based on Actions on Google policy, enabling Small Talk in its entirety will cause your action to be rejected. See Import the prebuilt agent for steps on how to import and select subsets of Small Talk features that comply with Action on Google's policy.' At the bottom, there is a 'Small Talk Customization Progress' section with two progress bars: 'About agent' and 'Courtesy', both at 0%.

## 8) Test your agent from Telegram

- Launch telegram and add your bot. You can search for your bot by adding a @ to the username you used in step 3. *\*same as how we searched for BotFather*
- Click start to start a conversation with the bot.
- Start asking your bot!

*E.g. Do I qualify for Merdeka? How to appeal? Where can I collect my token?*

### Activity wrap-up:

We learn how to

- ☐ Create a Dialogflow agent.
- ☐ Create a Dialogflow knowledge base
- ☐ Use Small Talk to add personality to agent
- ☐ Integrate Telegram to agent

## Activity 2 – Training an Image Classifier

In this activity, we will:

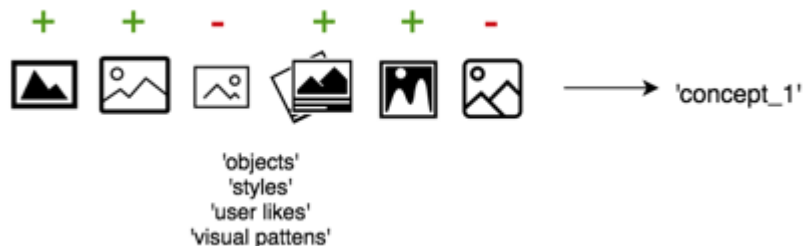
- ☐ Learn how to build a classifier through Clarifai AI service.
- ☐ Learn how to train a (supervised learning) image classification model
- ☐ Learn to evaluate the model
- ☐ Deploy your model to a chatbot

### NOTE for on-campus training:

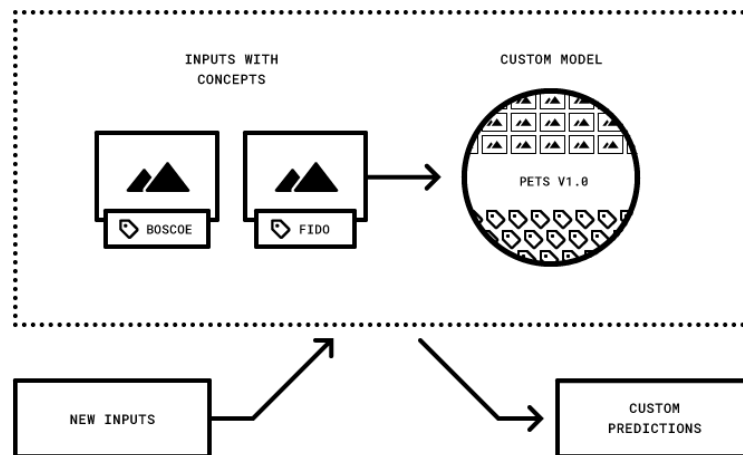
- ☐ **Some security and network settings may disable proper registration on Clarifai. Please check that you are using Guest@RP wireless network!**

### Custom Training

Custom Training is about teaching computers to see the world in a way that is specific to your own content and context. You can think of Custom Training as a series of inputs where you ultimately teach a neural network what **concept\_1** is and what it is not. We define the training data, taxonomy (the model's 'concepts', this is equivalent to “labelling”) and other performance characteristics.



If your content is visually distinct and easy to identify, 25-50 positive examples per concept will provide robust and accurate predictions. A **'concept' is synonymous with 'tag', 'category' or 'keyword'**. Concepts are your business' world view as to what an object, visual pattern, or style may represent.



What makes for a well-built concept?

- **Accurate labels.** Mis-labelled images introduces noise into your model and can lead to weak or confusing predictions.
- **Balanced training data.** Skewed training sets where several concepts have 5-20x as many positive training images as others may affect model performance.
- **Matching training and prediction context.** It's crucial that your training images for your concepts resemble the conditions and context of imagery you'll be making predictions on.

As an example, training a flower identification model solely with stock photography and then attempting to predict on user generated smartphone photos will not be ideal.

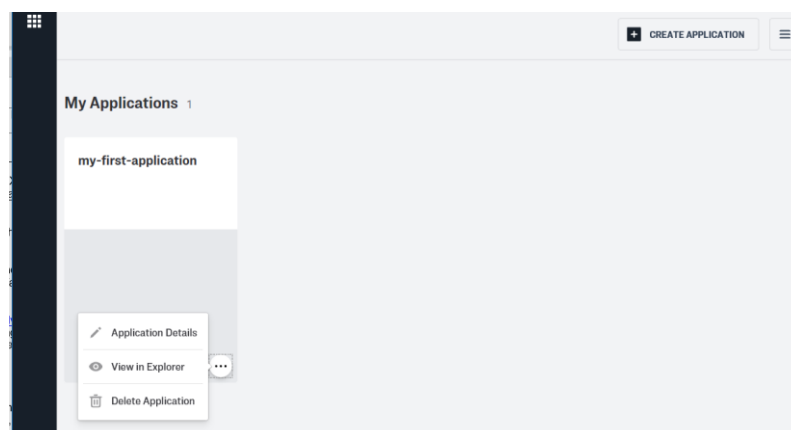
## 1. Create an account with Clarifai

- a) Sign up for an account at clarifai.com (<https://portal.clarifai.com/signup>)

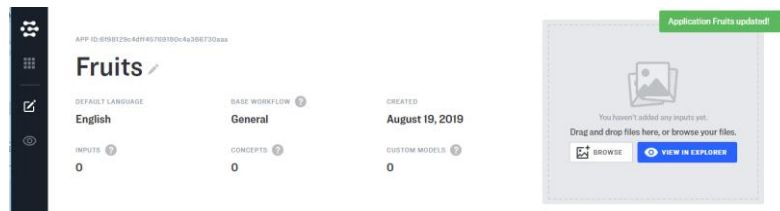
## 2. Edit your application

Operations are tied to an account and application, and any model that you create and add images to will be contained within a specific application. By default, you'll have an application in your account already so let's change that one's name to whatever you'd like. We are going to name ours "Fruits" for practical reasons

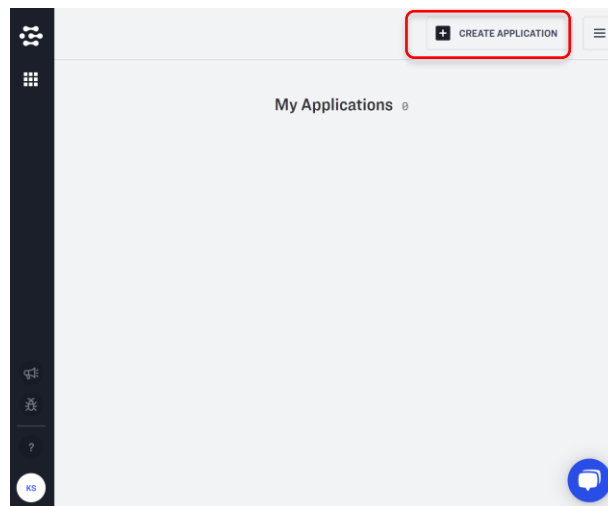
- a) Click on the details button of the default application and select Application Details.



- b) Change the application name to Fruits.



- c) Alternatively, you can also use “Create New Application” to create a new application and fill in the details as shown below.



### Create an application

Create your application here!

APP-ID ?

Fruits

DISPLAY NAME (OPTIONAL)

short descriptive name

DESCRIPTION (OPTIONAL)

short description about your application

DEFAULT LANGUAGE

English (en)

BASE WORKFLOW ?

General

Note: an API Key will be created automatically for this application.

CANCEL CREATE

For non-locked down laptops, you can perform step 3 – 9. If you are using a locked-down and cannot access the URL in step 3, skip to step 10.

3. Copy the API key from the Application created: [https://cognimate.me:2635/vision\\_home](https://cognimate.me:2635/vision_home)


## API Keys ?

[DETAILS](#)

Test2-all-scopes

96fdf30985094e5b899c78b3ba0e5486

Sep 3, 2020

4. Give your project a name and click on the “Set” button.

### What is your project name?

5. Paste the API key that you had copied earlier into the empty field and click on the “Set” button.

### What is your Clarifai key? ?

[How-to guide](#) [Sign up & get keys](#)


6. Key in the category of item that you want to train and click on “Add Category”.

### What categories should your model have? ?

7. Drag and drop 10 or more images of the item of choice. (Suggest to have two or more categories)

## What categories should your model have?

Orange



Drag and drop 10 or more images


Add Category

8. Click on “Train Model” once you the images are uploaded.


Train Model

9. Drag and drop an image of choice from the “test\_image” folder which you had downloaded earlier followed by clicking on the “Predict” button.

## Upload an image to test your model



Drop images here, or click to choose



28.4 KB

— or —

Predict

Category: Orange  
Confidence score: 100.00%

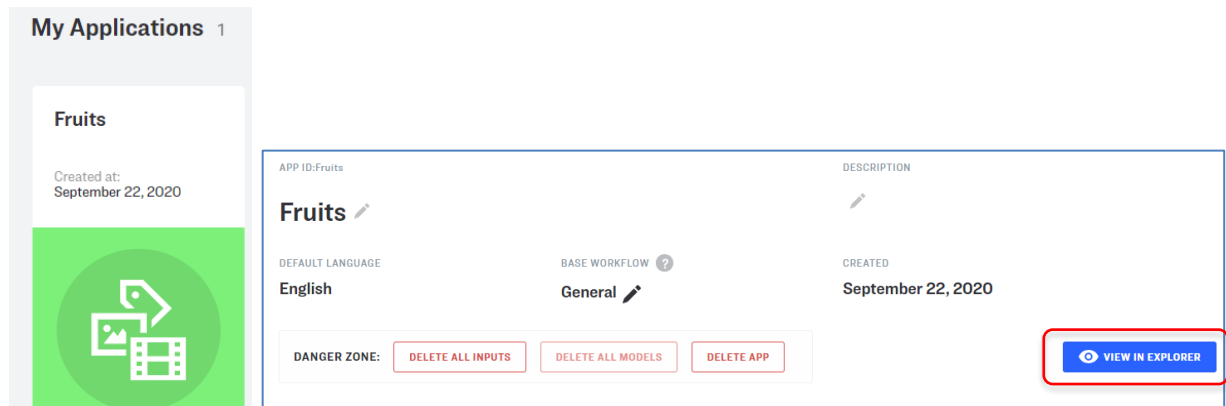
\*The predicted category will show together with the confidence level.


^If you have done the above, please skip to step 15.

For locked down laptops, proceed from this step onwards.

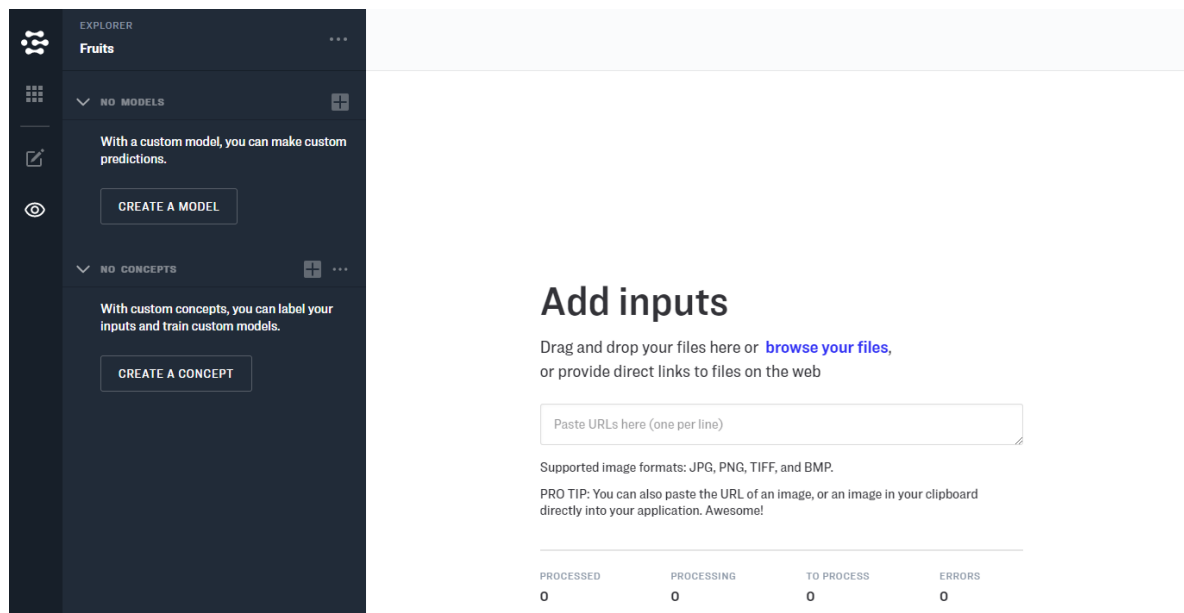
### 10. Add images to your application:

- Select your created **Application** and click on **VIEW IN EXPLORER** button to start adding images to your application.



- Click  on the top right and **BROWSE FILES**.

Custom models are built by training on your own data, and they will be able to make predictions specific to your own unique content and context.



- Start uploading your images, say, those banana images.
- A pop-up shown below appears. Click **X** to close the window. We will perform the configurations later.



✕

📌 Add or Assign Concepts

Select Custom Concepts

{ } Add Metadata

```
JSON: {"key": "value"}
```

📍 Add Geolocation

Latitude

Longitude

i.e: 51.477928

i.e: -0.001545

Add Data To Inputs

- e. It may take a while to upload the images. Once uploaded, you will see tick mark with the uploaded images.

Drag & Drop your files here

BROWSE FILES

Supported image formats: JPG, PNG, TIFF, BMP, WEBP, CSV, and TSV

Or provide direct links to files on the web

PRO TIP: You can also paste an asset URL, or an asset in your clipboard directly into your application. Awesome!

PROCESSED

PROCESSING

TO PROCESS

ERRORS


30


0


0


0


Please don't close this page while upload is in progress.
















































































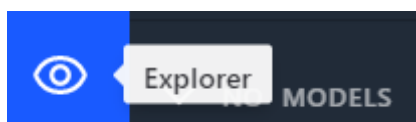






\*Note: It may take a while to upload the images. Wait until all images are PROCESSED and the rest are 0.

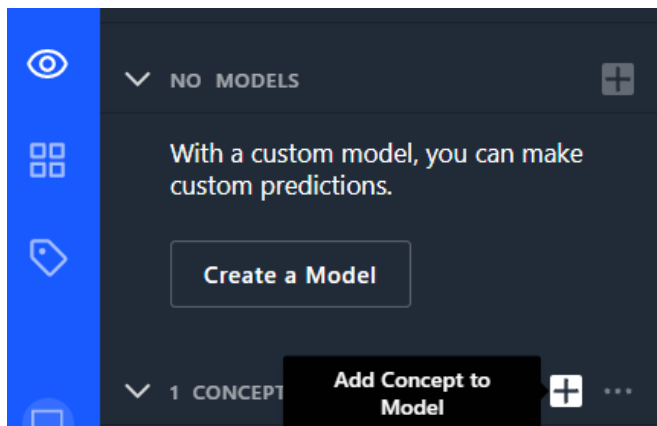
f. On the left navigational menu, select **Explorer** icon. You will see the uploaded images on the right panel.

The image shows a dark blue sidebar with a white eye icon and the word 'Explorer' in white text. To the right of this, the word 'MODELS' is visible in a lighter blue font.

Version 1.5

## 11. Labelling your images using Concepts.

- a) Click + on **CONCEPT** to **Add Concept to Model**.

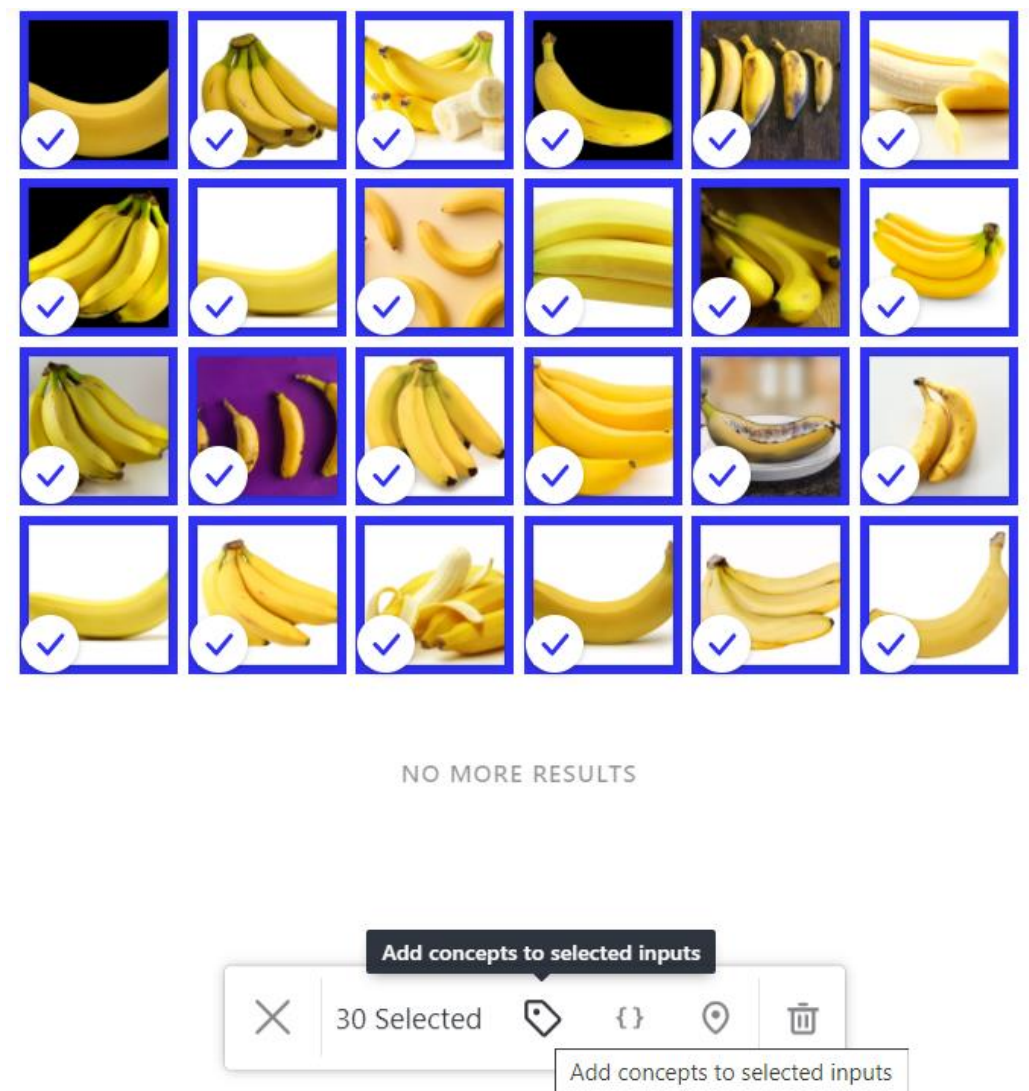


- b) Type in **bananas** for your new concept name, click **add** and **done**.

Create new concept

- c) From navigation menu, select **Explorer**.  
d) Select all the banana images and click **Add concept to selected inputs** from the bottom menu.



- e) Select **bananas** which you have just created. This will label all the images of bananas as **bananas** and click **Label 30 inputs with selected concepts**.

## Label selected inputs (30)

Filter Concepts

bananas

Include Concepts

1.



bananas

2.

Label 30 inputs with selected concepts

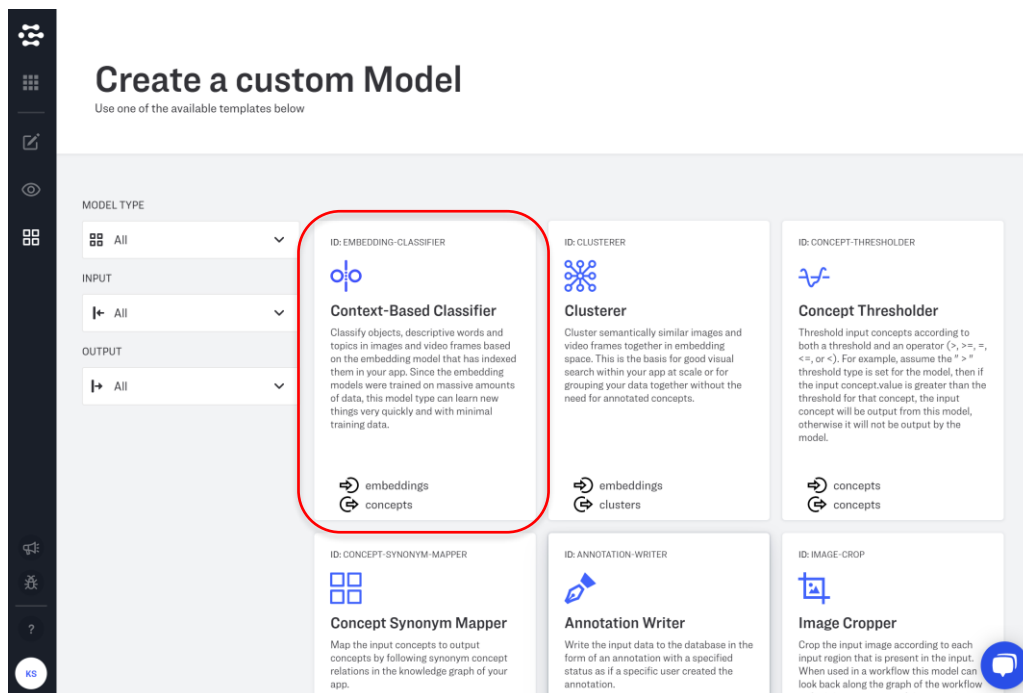
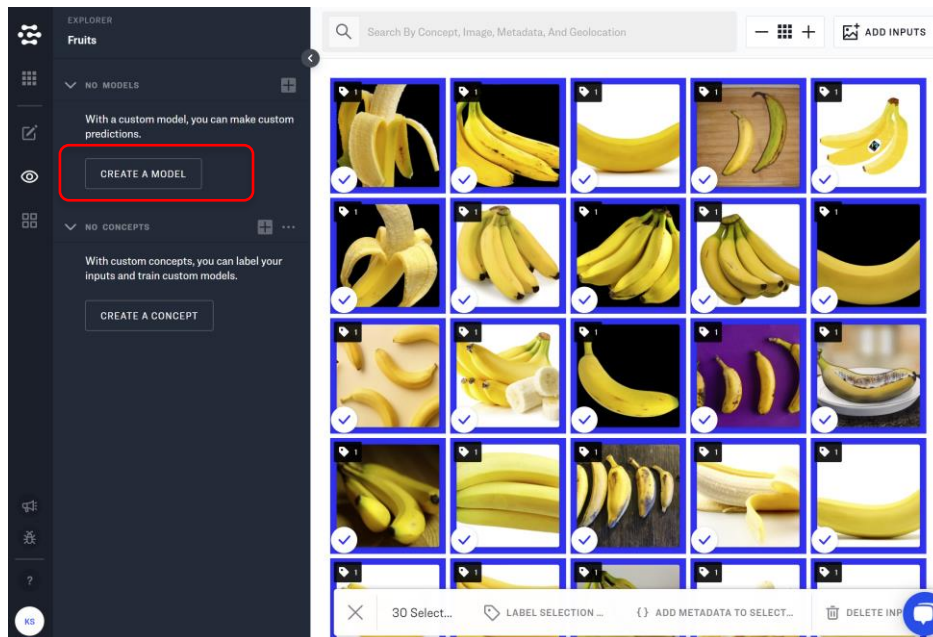
- f) To verify the labelled images, select **bananas** from **CONCEPT** via the navigation menu.

## 12. Add more images and concepts

Repeat necessary steps to add and label images using concepts, for the rest of the fruits folders (except test\_images)

## 13. Create and Train the model

Once all of your images are labelled, let's go ahead and create your first model by clicking **CREATE A MODEL**.



- a) Click on **Context-based Classifier**. Enter a name (*avoid space within name*) for the model and other details as shown. Click on **CREATE MODEL**.

**Create a Context-Based Classifier Model**

Classify objects, descriptive words and topics in images and video frames based on the embedding model that has indexed them in your app. Since the embedding models were trained on massive amounts of data, this model type can learn new things very quickly and with minimal training data.

MODEL ID (OPTIONAL) ?

user-defined ID

DISPLAY NAME ?

Fruits

OUTPUT\_INFO.DATA.CONCEPTS \* ?

CONCEPT ID	CONCEPT NAME
coconut	coconut
banana	banana

Concept Name

OUTPUT\_INFO.OUTPUT\_CONFIG.CONCEPTS\_MUTUALLY\_EXCLUSIVE ?

If you expect multiple concepts for this model to be present per image, set 'Concepts Mutually Exclusive' to false.

OUTPUT\_INFO.OUTPUT\_CONFIG.CLOSED\_ENVIRONMENT ?

If you will be training with images that do not contain any concepts for this model, set 'Closed Environment' to false.

OUTPUT\_INFO.OUTPUT\_CONFIG.EMBED\_MODEL\_VERSION\_ID ?

bb186755eda04f9cbb6fe32e816be104

**CREATE MODEL**

Let's go ahead and train your first model. You can either do this via the little 3-dot menu next to the model name or you can click on the model name and then click the Train Model button on the ensuing page.

- b) Click on the application (ie Fruits). You should see the following. Click on **TRAIN MODEL**.

*\* you will see at the bottom right screen on the status of model being trained and completed.*

EXPLORER

Fruits

MODELS

Fruits

CONCEPTS

banana

coconut

MODEL ID: Fruits

EDIT MODEL DETAILS

CREATED: Jul 1, 2020 at 11:18 AM

ACTIVE MODEL

**TRAIN MODEL**

Concepts 2 Versions

VERSION ID	STATUS	CONCEPTS	INPUTS	ROC AUC	METRICS	CREATED
684a1eb9ec1d4dac8e20d180c57d2168	Model not yet trained	2				Jul 1, 2020 11:18am

#### 14. Evaluate your model

- a) Click the **Explorer**, you should be able to see the model name. **Refresh** the webpage if the model name isn't there.

- b) Click on the model name that appears on the left panel to bring you to the screen above.
- c) Click on **Versions**. It will take you to the following screen:

MODEL ID: Fruits

CREATED: Aug 24, 2019 at 01:38 PM

ACTIVE MODEL TRAIN MODEL

EDIT MODEL DETAILS

Concepts 8 Versions

VERSION ID	STATUS	CONCEPTS	INPUTS	ROC AUC	METRICS	CREATED
4fd32fe4fb442d289bd8cd39af575f7	Model trained successfully	8	288		EVALUATE	Aug 24, 2019 2:04pm
2cb2122a05da42af835fa1d81293583d	Model not yet trained	8				Aug 24, 2019 2:01pm
e657b1c2ae1149e29da9190132f41499	Model not yet trained	7				Aug 24, 2019 2:00pm
ec01cf662cc4483a890af3853727d8a	Model not yet trained	6				Aug 24, 2019 1:59pm
0b82be31828c4fa2b5067eb938599acf	Model not yet trained	5				Aug 24, 2019 1:57pm
fe9dd420bfa944ea0d0a7457e9c9150	Model not yet trained	4				Aug 24, 2019 1:55pm
959924fa385e4c2fa972825fd3537a3e	Model not yet trained	3				Aug 24, 2019 1:53pm

- d) To evaluate your model, click the **evaluate** option under Metrics. This will take a short amount of time depending on the number of images added to your model. Our simple model should be evaluated in seconds. Once the evaluation is completed, the **"Evaluate"** option will be changed to a **"View"** option. Click it, and you will see the evaluation results.

Concepts 8 Versions

▼ Evaluation Summary Table

CONCEPT	K-SPLIT AVG ACCURACY SCORE (ROC AUC)	I SPLIT						
		TOTAL LABELED	TOTAL PREDICTED	TRUE POSITIVES	FALSE NEGATIVES	FALSE POSITIVES	RECALL RATE	PRECISION RATE
durian	1.000	6	6	6	0	0	1.000	1.000
apple	1.000	7	8	7	0	1	1.000	0.875
strewberry	1.000	9	9	9	0	0	1.000	1.000
banana	1.000	7	7	7	0	0	1.000	1.000
dragon fruits	1.000	8	8	8	0	0	1.000	1.000
pineapple	1.000	5	5	5	0	0	1.000	1.000
orange	1.000	9	9	9	0	0	1.000	1.000
coconut	1.000	7	7	7	0	0	1.000	1.000
TOTAL	AVG: 1.000	58	59	58	0	1	AVG: 1.000	AVG: 0.984

► Concept by Concept Results

▼ Co-occurrence

Version Details

ID: 4fd32fe4fb442d289bd8cd39af575f7

CREATED AT: Aug 24, 2019 2:04 pm

UNIQUE CONCEPTS: 8

CONCEPTS MUTUALLY EXCLUSIVE: false

CLOSED ENVIRONMENT: false

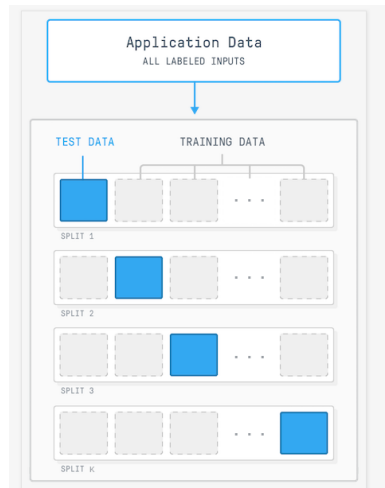
Selection Details

IMAGES LABELED: apple

PREDICTION PROBABILITY FOR: apple

	1.0000
	1.0000
	1.0000
	1.0000
	1.0000
	0.9998
	0.9992

- e) Evaluation results will be categorized under 4 headings: **Evaluation Summary Table**, **Concept by Concept Results**, **Co-occurrence**, **Precision-Recall and ROC Curves**. Clarifai does the evaluation using a method called K-split cross-validation on the data.



Clarifai uses a random subset of our input data as test data to test against a new model with the remaining data. Then, it predicts the test data against the new model. After that, it compares the predicted labels with the real labels. After performing this multiple times, Clarifai gives each concept a score.

There is a value called **probability threshold**, which determines the point at which concepts will be classified as either positive or negative. For example, an image is counted as belonging to a particular concept, such as a pineapple, only if its prediction probability of that image for the pineapple is higher than the threshold value. The default threshold value is 0.5. You can change it as you want.

- f) Click on the “i” icon to see the evaluation details of a particular concept.

Concepts 8 Versions

▼ Evaluation Summary Table ⓘ

Model Accuracy Score (ROC AUC MAC AVG):1

Current Prediction Threshold is 0.5. This means an input 'counts' as a predicted concept if the prediction probability for that concept is greater than or equal to 0.5.

CONCEPT	K-SPLIT AVG	TOTAL LABELED	TOTAL PREDICTED	TRUE POSITIVES	FALSE NEGATIVES	FALSE POSITIVES	RECALL RATE	PRECISION RATE
durian	1.000	6	6	6	0	0	1.000	1.000
apple	1.000	7	8	7	0	1	1.000	0.875

Of the 7 images actually labeled **apple**:

- True Positive: 7 were predicted as **apple** with probability greater than or equal to 0.5
- False Negative: 0 were predicted as **apple** with prediction probability less than 0.5
- Recall Rate: 100% (=7/7) of the images actually labeled **apple** were predicted as **apple**.

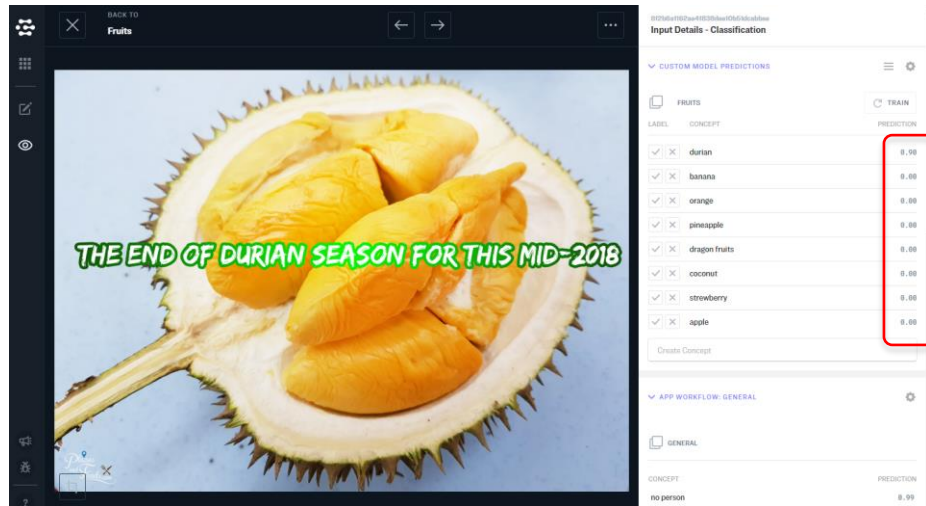
Of the 8 images predicted as **apple** with prediction probability greater than or equal to 0.5:

- True Positive: 7 were labeled as **apple**.
- False Positive: 1 were not labeled as **apple**
- Precision Rate: 87.5% (=7/8) of the images predicted as **apple** were labeled as **apple**.

## 15. Test your model (Upload some external images and see how the model is performing)

- Click on the application, **ADD INPUTS** to upload a test image. (for this exercise, use an image from the test\_images directory)
- Click on the test image (via **Explorer**) to test the image classification.



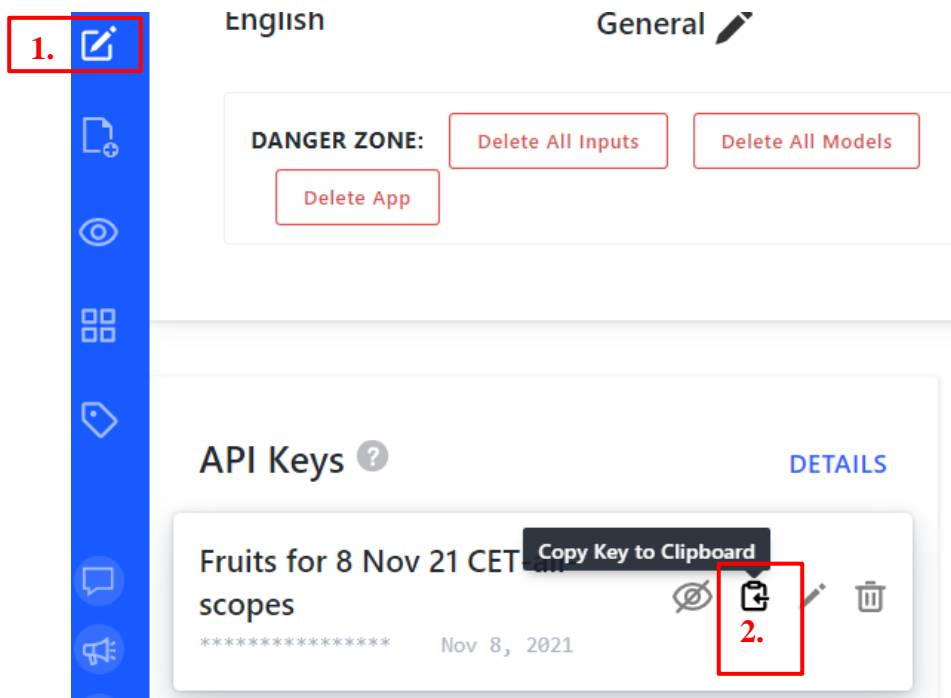


## 16. Adding negative examples when the model is getting confused (optional)

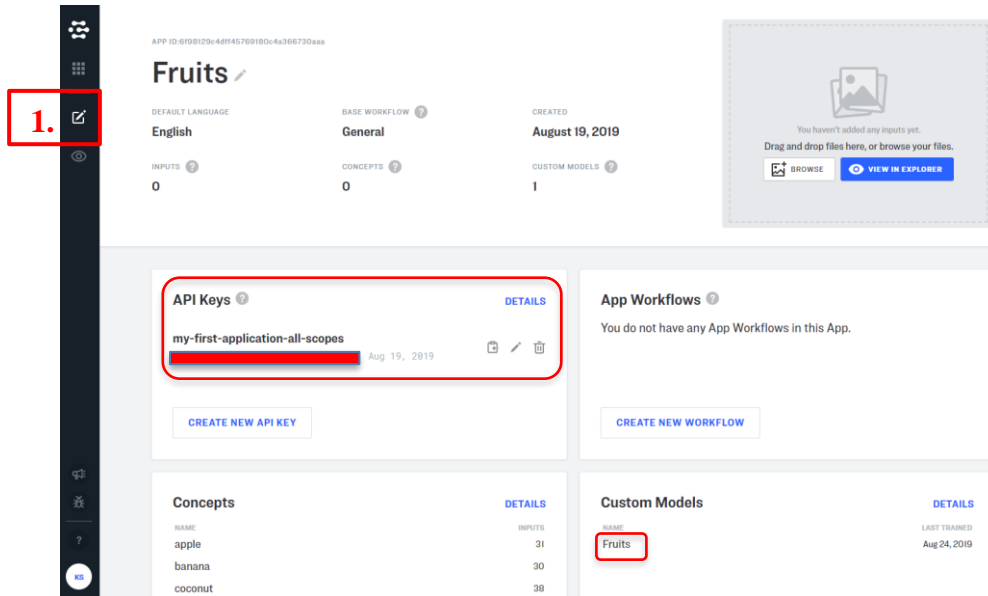
Up until this point we have only added positive examples to the model (e.g. saying "this is definitely concept X"), but we haven't added any negatives that say the opposite (i.e. "this is not concept X"). A robust and well performing concept is typically made up of both positive and negative examples with a 3 or 4:1 ratio, respectively, but adding too many will be counterproductive so be careful. If you see in one instance that you're getting a false positive for "apple", for example, you may want to teach the system that it's wrong.

## 17. Model Deployment and usage

- Once you trained a model, an API key is automatically created for you. To use this model in an application, you will need this API key and the model name. Click on **Application Details** and **Copy Key to Clipboard** to get this information from the screen.







## 18. Use the model via a web app

- Launch your web browser and go to <http://kwseow.pythonanywhere.com/forms>. This is a demo site created to utilise your model to perform image classification.
- Key in your Clarifai API key and the name of the model you trained in the previous step.
- Select a photo you want to test with and click on submit. After a short while you should see the result of the classification of your model. See the following screen capture:

### AI4E Clarifai image classification demo

Clarifai API Key

Clarifai custom model name

No file chosen

```
Success! durian|0.9999454
pineapple|0.0007022619
coconut|3.5732985e-05
dragon fruits|1.692772e-05
orange|6.3478947e-06
banana|3.1590462e-06
strewberry|1.7881393e-07
apple|2.9802322e-08
```

Using model:  
Fruits|Fruits



d) The following code snippet provide gives you an ideal of how this is done.

```

01 @app.route("/forms", methods=['GET', 'POST'])
02 def myforms():
03     form = ReusableForm(request.form)
04
05     #print(form.errors)
06     if request.method == 'POST':
07         form = ReusableForm()
08
09         if form.validate_on_submit():
10             #write_to_disk(name, surname, email)
11             f = form.photo.data
12             filename = secure_filename(f.filename)
13             f.save(os.path.join('./mysite/static/photos', filename))
14             try:
15                 #api_key = ""
16                 api_key = form.api_key.data
17                 os.environ["CLARIFAI_API_KEY"] = api_key
18                 clarifai_app = ClarifaiApp()
19                 #custom model
20                 #model_name="Fruits"
21                 model_name=form.model_name.data
22                 model = clarifai_app.models.get(model_name=model_name)
23                 response = model.predict_by_filename(os.path.join('./mysite/static/photos',
24 filename ))
25                 msg = ""
26                 for concept in response['outputs'][0]['data']['concepts']:
27                     msg += "%s\n"%(concept['name'],concept['value'])
28                 tmp_model = response['outputs'][0]['model']
29                 msg += "\nUsing model:\n%s\n"%(tmp_model['id'],tmp_model['name'])
30                 msg = msg.replace('\n', '<br>')
31             except ApiError as e:
32                 msg = 'Error status code: %d\n' % e.error_code
33                 msg += 'Error description: %s\n' % e.error_desc
34                 if e.error_details:
35                     msg += 'Error details: %s' % e.error_details
36
37                 if e.error_code == 21200:
38                     api_key=form.api_key.data
39                     os.environ["CLARIFAI_API_KEY"] = api_key
40                     clarifai_app = ClarifaiApp()
41                     msg += "\nAvailable models:"
42                     for model in clarifai_app.models.get_all():
43                         msg += "\n%s"%model.model_name
44
45                     msg = msg.replace('\n', '<br>')
46
47             #flash('Hello: {}'.format(filename))
48             flash('{}{}'.format(msg))
49             #filename = secure_filename(form.file.data.filename)
50             #form.file.data.save('uploads/' + filename)
51             #flash('Hello: {} {} {}'.format(name, surname,filename))
    
```

```
47  
48         else:  
49             flash('Error: All Fields are Required')  
50  
51     return render_template('form.html', form=form)
```

Activity wrap-up:

We learn how to:

- ☐ Train and evaluate an image classifier
- ☐ Deploy the model for use in a chatbot and a web browser