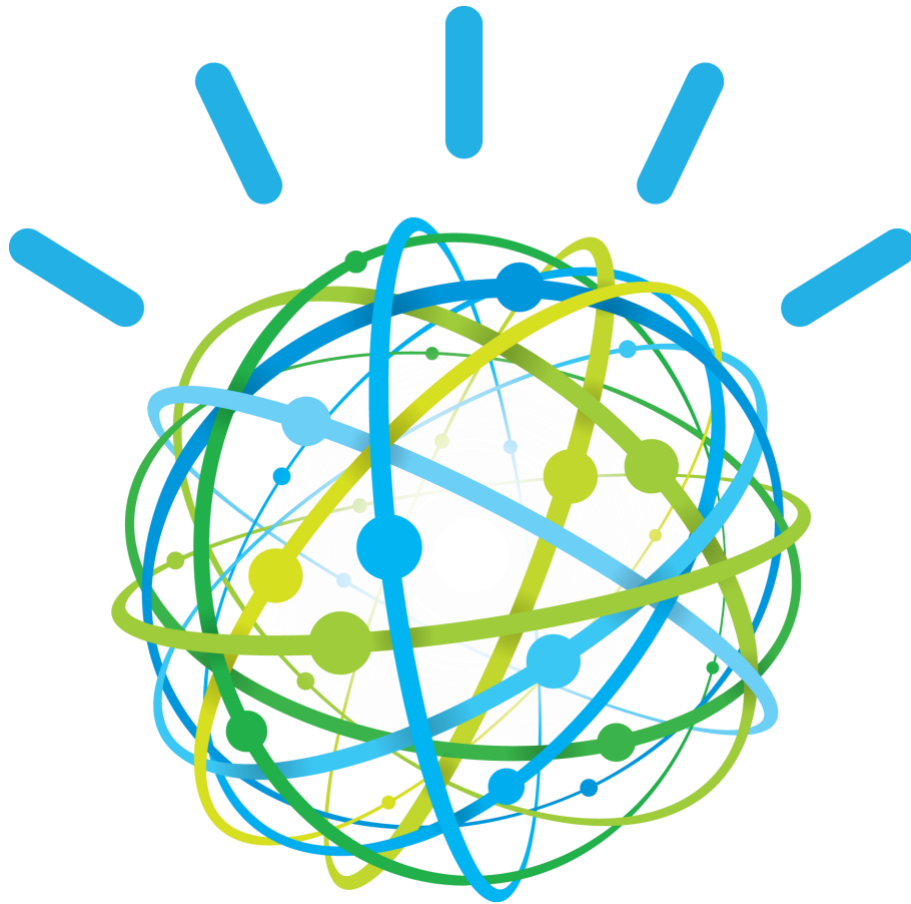


Session 6: Improving
IBM Watson Assistant



Lab Instructions

Laurent Vincent

Content

Let's get started	3
1. Overview	3
2. Objectives	3
3. Prerequisites.....	3
4. Scenario	4
5. What to expect when you are done	4
Improving understanding	5
6. Overview page	5
7. User conversations page.....	8
Conversation Sample Application	9
8. Install locally your test application.....	9
9. Add a deployment ID.....	12
10. Start your local application	13
11. Test your local application	14
Sending utterances and generating logs with Watson Studio	17
12. Create a Watson Studio instance	17
13. Create Watson Studio resources.....	18
Improve Conversation	23
14. Navigate on Overview page.....	23
15. Create a test-experimental workspace	23
16. Selecting data source / Deployment ID	24
17. Adding an entity value or synonym.....	25
18. Correcting an intent.....	27
19. Irrelevant input	30
20. Test your conversation updates.....	31
Test your improvements with Watson Studio	32
21. Prerequisite : create a cloudant service.....	32
22. Create Watson Studio resources.....	34

Let's get started

1. Overview

The [IBM Watson Developer Cloud](#) (WDC) offers a variety of services for developing cognitive applications. Each Watson service provides a Representational State Transfer (REST) Application Programming Interface (API) for interacting with the service. Some services, such as the Speech to Text service, provide additional interfaces.

The [Watson Assistant](#) service combines several cognitive techniques to help you build and train a bot - defining intents and entities and crafting dialog to simulate conversation. The system can then be further refined with supplementary technologies to make the system more human-like or to give it a higher chance of returning the right answer. Watson Conversation allows you to deploy a range of bots via many channels, from simple, narrowly focused bots to much more sophisticated, full-blown virtual agents across mobile devices, messaging platforms like Slack, or even through a physical robot.

The **illustrating screenshots** provided in this lab guide could be slightly different from what you see in the Watson Assistant service interface that you are using. If there are colour or wording differences, it is because there have been updates to the service since the lab guide was created.

2. Objectives

Watson Conversation Service provides features to analyse its usages.

In this lab, you will:

- Learn how to create a Node.js application running locally
- Learn how to improve the Watson Conversation service using Node.js
- Learn how to leverage Watson studio to test your service

3. Prerequisites

You must have install on your workstation Node.js and Cloud Foundry as describes in the prerequisite document

- Session 5 – Building a dialog
- The instructor provided you the link to get labs content. You may download each file individually.

IBM Cloud URLs per location:

Location	URL
US	https://console.ng.bluemix.net/
UK	https://console.eu-gb.bluemix.net/
Sidney	https://console.au-syd.bluemix.net/
Germany	https://console.eu-de.bluemix.net/

4. Scenario

Use case: A Hotel Concierge Virtual assistant that is accessed from the guest room and the hotel lobby.

End-users: Hotel customers

5. What to expect when you are done

At the end of session, you should have

1. one Workspace for production
2. one Workspace for test
3. a simple node.js application to start testing your conversation
4. a Watson studio service with 2 notebooks
5. the right understanding of the improve capabilities.

Improving understanding

The Improve component of the Conversation service provides a history of conversations with users. You can use this history to improve your bot's understanding of users' inputs.

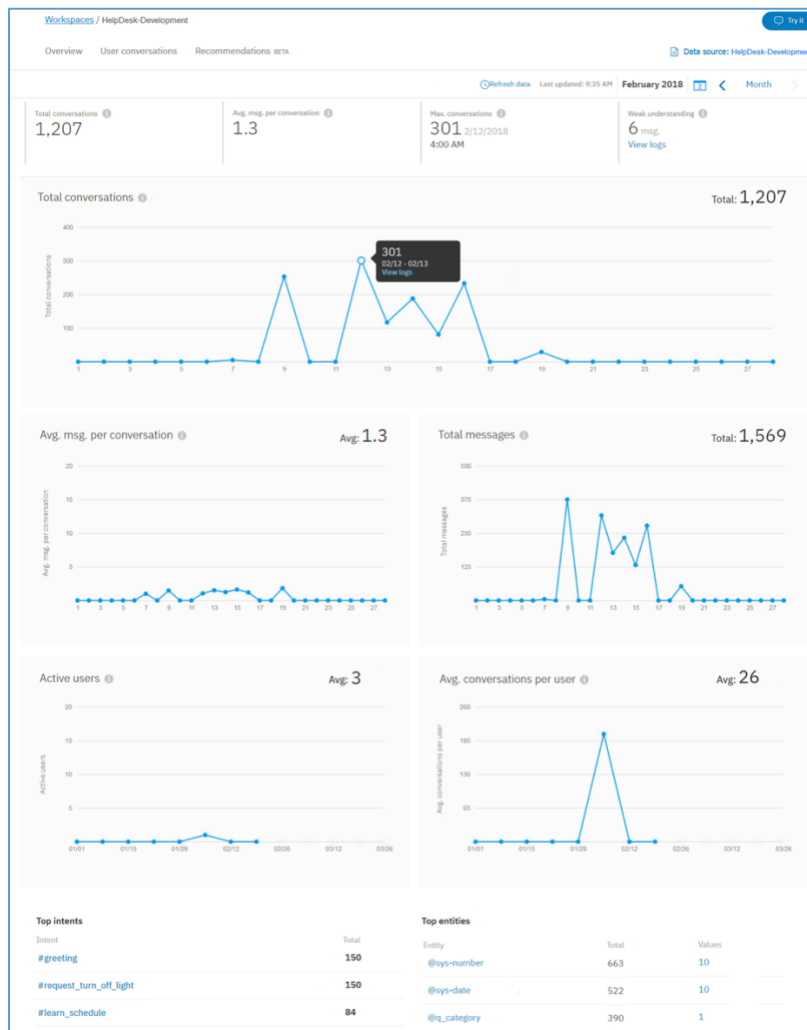
While you develop your workspace, you use the **Try it out** panel to verify that it recognizes the correct intents and entities in test inputs, and make corrections as needed. In the **Improve** panel, you can view information about actual conversations with your users and make similar corrections to improve the accuracy with which intents and entities are recognized

1. Click on the hamburger menu (top left) and click on **Improve**

The **User Conversation** page of the Improve component provides a summary of interactions with your bot, such as Statistics and chat logs for the last 90 days are provided, with the newest first. You can view the traffic for a given time period, as well as the intents and entities that were recognized in user conversations. These statistics represent external traffic that has interacted with your bot; **they do not include interactions that use the Try it out panel in the tool.**

6. Overview page

1. Click on **overview** tab



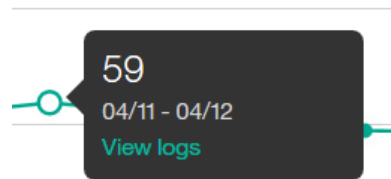
Use the time period control to choose the period for which data is displayed. This control affects all data shown on the page: not just the number of conversations displayed in the graph, but also the statistics displayed along with the graph, and the lists of top intents and entities.

04/09/2017 - 04/15/2017 < Week >

You can choose whether to view data for a single day, a week, a month, a quarter, or a year. In each case, the data points on the graph adjust to an appropriate measurement period. For example, when viewing a graph for a day, the data is presented in hourly values, but when viewing a graph for a week, the data is shown by day. A week always runs from Sunday through Saturday. You cannot create custom time periods, such as a week that runs from Thursday to the following Wednesday, or a month that begins on any date other than the first.

You can select **View logs** to open the **User Conversations** page with the date range filtered to match the time period that you have selected for the **Overview** page. Depending on your plan and the date range that you selected, it is possible that the **User Conversations** page will show no data.

While viewing the graph, you can click on an individual data point to see the numeric value, as shown here:



You can use the **View logs** link to open the **User Conversations** page with the date range filtered to match the data point.

In addition to the graph, four statistics related to the displayed data are shown:

1. The **total number of conversations** that took place during this time period
2. The **average messages per conversation** is the total messages divided by the total conversations during the selected date range.
3. The **maximum number of conversations** for a single data point within the time period
4. The **number of messages** not managed correctly.

You also see the intents and entities that were recognized most often during the specified time period. By default you see the top three of each, but you can change that to a larger number.

Intents are shown in a simple list. In addition to seeing the number of times an intent was recognized, you can click on the intent name to open the **User Conversations** page with the date range filtered to match the data you are viewing, and the intent filtered to match the selected intent.

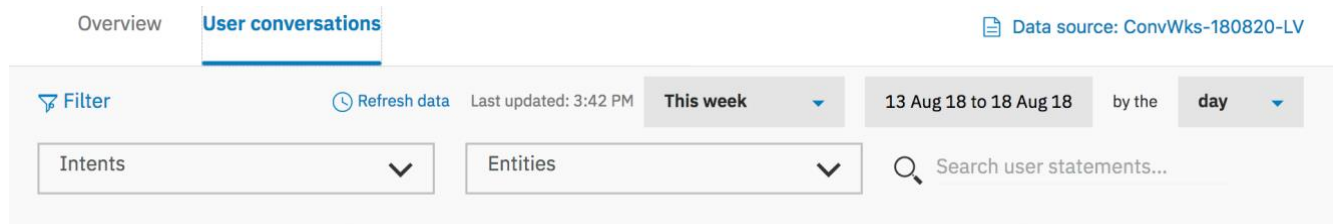
Entities are shown in a simple list. Click on it to open the **User Conversations** page with the date range filtered to match the data you are viewing and the entity filtered to match the selected entity.

Top intents		Top entities
Intent	Total	Entity
#greeting	160	@sys-date
#exit	37	@restaurant
#repeat_request	27	@sys-number

The page shows when the data that it displays was last updated. You can select **Refresh data**, at the top of the page, if you think that newer data might be available.

7. User conversations page

As you didn't do any interaction with external application, the page should be empty:



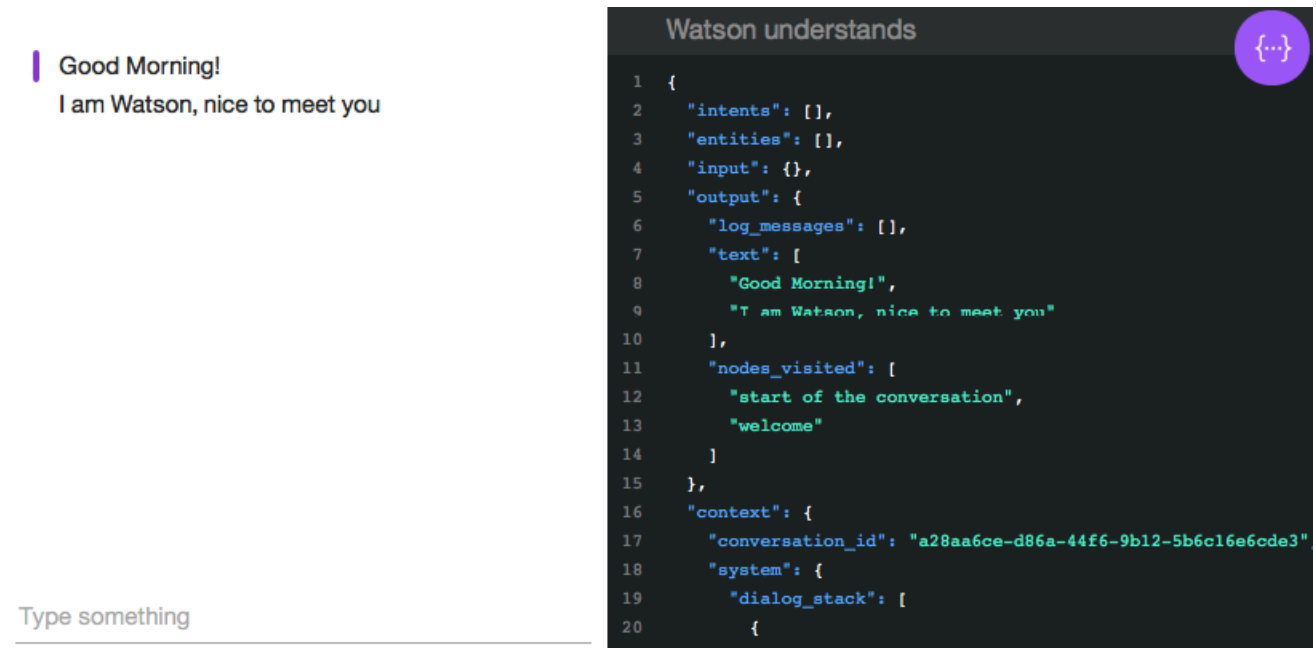
The screenshot shows the 'User conversations' page interface. At the top, there are two tabs: 'Overview' and 'User conversations', with 'User conversations' being the active tab. To the right, it says 'Data source: ConvWks-180820-LV'. Below the tabs is a filter bar containing a 'Filter' icon, a 'Refresh data' button, and the text 'Last updated: 3:42 PM'. To the right of this are three dropdown menus: 'This week', '13 Aug 18 to 18 Aug 18', and 'by the day'. Below the filter bar are two input fields: 'Intents' and 'Entities', each with a dropdown arrow. To the right of these is a search bar with a magnifying glass icon and the text 'Search user statements...'.

You can also filter the chat logs by:

- **Search user statements...** - Type a word in the search bar. This searches the user's input, but not your bot's reply.
- **Intents** - Select the drop-down menu and type an intent in the input field, or choose from the populated list. You can select more than one intent, which filters the results using any of the selected intents.
- **Entities** - Select the drop-down menu and type an entity name in the input field, or choose from the populated list. You can select more than one entity, which filters the results by any of the selected entities. If you filter by intent *and* entity, your results will include the messages that have both values.
- **Date range** - Select the drop-down menu and choose **Last 7 days**, **Last 30 days**, **Last 90 days (Default)**, or **Custom**. If you choose Custom, type the date range or select it from the calendar. Chat logs history is dependent on your service plan, as stated in the plan description.

Conversation Sample Application

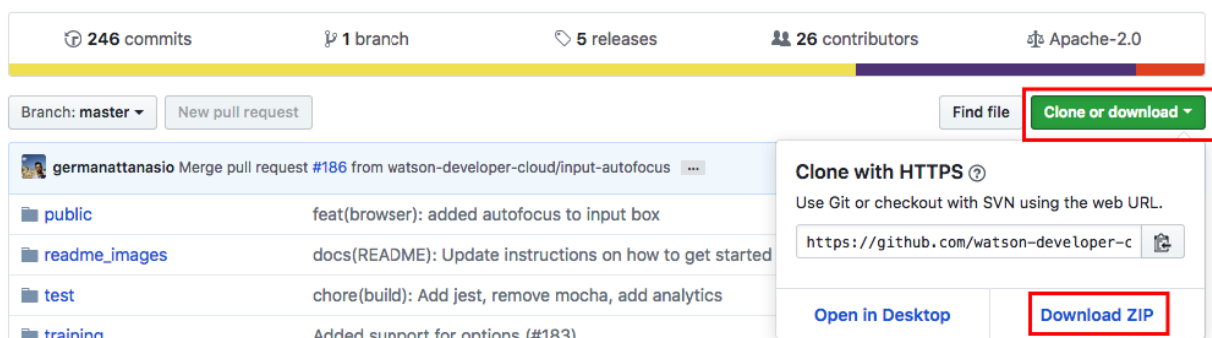
The Node.JS app you are going to install in your laptop is the conversation service in a simple chat interface. It can be used as a basis for testing, improving and building your own application.



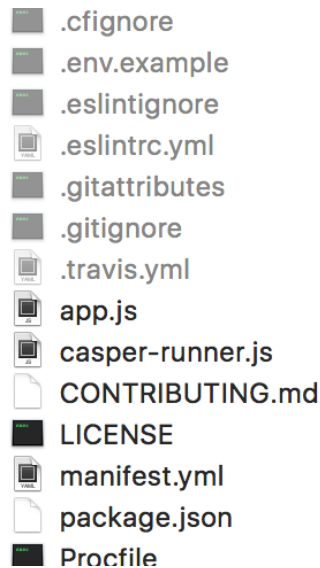
8. Install locally your test application

1. Use GitHub to clone the repository locally, Go to <https://github.com/watson-developer-cloud/assistant-simple>
2. Click on **Clone or download** button and **Download ZIP**

A simple sample application demonstrating the Watson Assistant api. <https://assistant-simple.ng.bluemix.net/>



3. Then select the right destination folder to save the **assistant-simple-master.zip** file.
4. Several extracted files are hidden such as .env.template file. If you don't see it in the unzipped folder conversation-simple-master, you must enable this option "show hidden file" on your laptop.



On IOS:

1. Open Terminal
2. Run the following script:

```
defaults write com.apple.Finder AppleShowAllFiles true  
killall Finder
```

If you want to switch it back just change the true to false.

On windows: update folder settings

5. Copy or rename the *.env.example* file to *.env*

On windows:

1. Open Terminal
2. Go to the location of your project

On windows:

3. Run the following script:

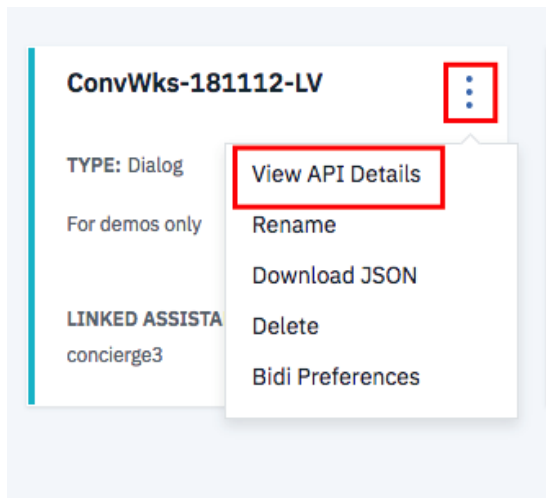
```
Copy .env.example .env
```

On IOS:

4. Run the following script:

```
Cp .env.example .env
```

- 6. Retrieve the credential of your skill. Go back to the **skills** tab
- 7. Click on 3 dots menu of your skill tile, click **View API Details**



then you get

Skill Details

Skill name: ConvWks-181112-LV

Skill ID: dac64476-3219-4c41-9589-6da020001097

Workspace ID: 58569bbf-6440-4b5d-8d4d-0797

Skill URL: https://gateway.watsonplatform.net/assistant/api/v1/workspaces/dac64476-3219-4c41-9589-6da020001097/message

Service Credentials

Credentials name: auto-generated-apikey-d5610

Username: apikey

Password: ax

here you will find all details such as Username, Password, Assistant ID required to leverage Watson conversation APIs.

- 8. Open **.env** file
- 9. Paste the Workspace, User and password values (without the <> marks) into the corresponding values.

```
# Environment variables
```

```
WORKSPACE_ID= <Workspace ID>
```

```
# You need to provide either username and password
```

```
ASSISTANT_USERNAME=apikey
```

```
ASSISTANT_PASSWORD=<password>
```

```
ASSISTANT_URL=https://gateway.watsonplatform.net/assistant/api
```

- 10. Save **.env** file

9. Add a deployment ID

For a Watson Assistant Service that includes multiple workspaces, it could be useful to use utterance data from one workspace to improve another workspace within that same service.

Note: If you are a Assistant Premium user, your premium instances can optionally be configured to allow access to log data from workspaces across your different premium instances.

As an example, you may have both a Production workspace and a Development workspace in your Assistant service. When working in the Development workspace, you can filter utterances based on the *Deployment ID* for the Production workspace, so that you are using Production workspace utterances to improve your Development workspace.

You might also want to filter data regarding the channel, environment or user group. To make this possible, you are going to add the deployment ID in the message you send to WCS. This is a property inside the metadata object in your context.

1. Edit app.js file to get this result

```
40 // Endpoint to be call from the client side
41 app.post('/api/message', function(req, res) {
42   var workspace = process.env.WORKSPACE_ID || '<workspace-id>';
43   if (!workspace || workspace === '<workspace-id>') {
44     return res.json({
45       'output': {
46         'text': 'The app has not been configured with a workspace'
47       }
48     });
49   }
50
51   var newcontext = req.body.context || {};
52   var newmetadata = {'deployment': 'ChatbotProduction'};
53   newcontext['metadata'] = newmetadata;
54
55   var payload = {
56     workspace_id: workspace,
57     // context: req.body.context || {},
58     context: newcontext || {},
59     input: req.body.input || {}
60   };
61 }
```

2. Look for the payload definition
3. Add the deployment ID definition, around the line 50

```
var newcontext = req.body.context || {};
var newmetadata = {'deployment': 'ChatbotProduction'};
newcontext['metadata'] = newmetadata;
```

4. Comment the line context: `req.body.context || {}` and add the code like this

```
// context: req.body.context || {},
context: newcontext || {},
```

5. Save and close the file

10. Start your local application

1. In a terminal/command window, issue a change directory command to navigate to a top-level directory “**assistant-simple-master**” that will contain your Node.js projects.

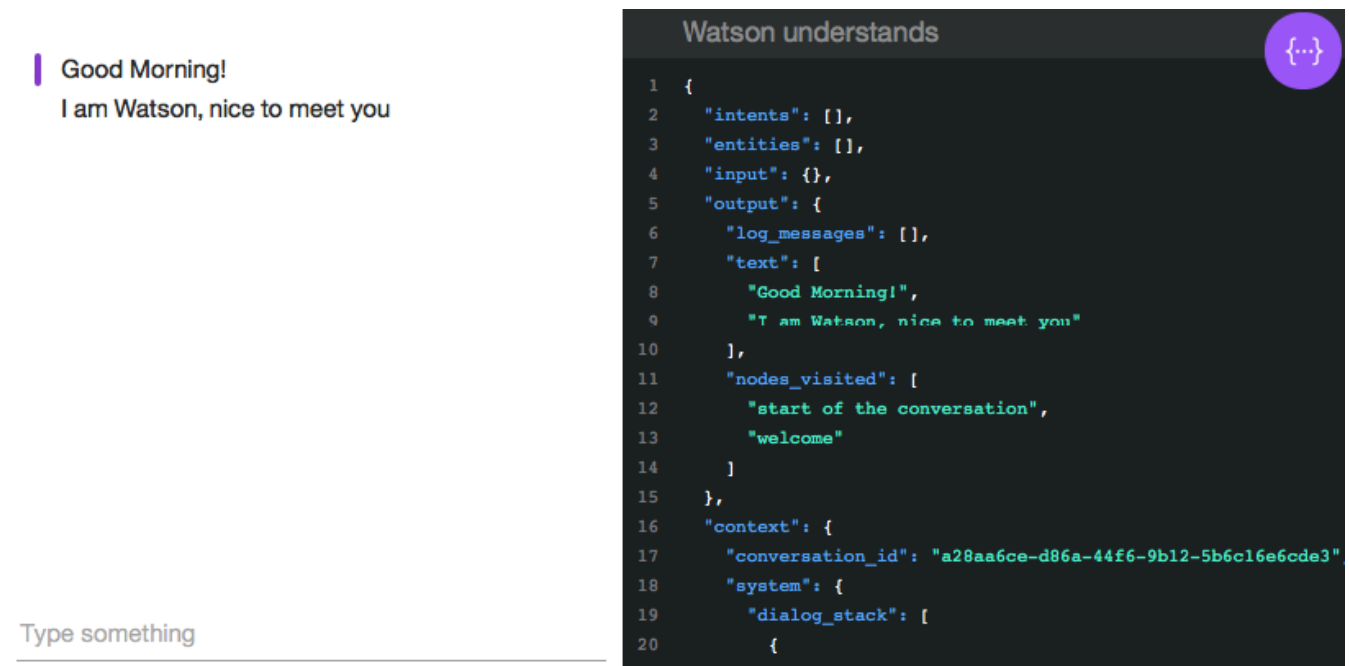
2. Install all dependencies of your demonstration application package, enter the command :

```
npm install
```

3. Start your application, enter the command:

```
npm start
```

4. Point your browser to <http://localhost:3000> to try out the app.



The screenshot shows a chat interface on the left and a JSON response on the right. The chat interface has a purple header bar with a menu icon. Below it, the text "Good Morning!" is displayed in a purple font, followed by "I am Watson, nice to meet you" in a grey font. At the bottom, there is a text input field with the placeholder "Type something". The JSON response on the right is titled "Watson understands" and shows a structured object with fields for intents, entities, input, output, log_messages, text, nodes_visited, context, and system. The text field contains the response "Good Morning!" and "I am Watson, nice to meet you".

The chat interface is on the left, and the JSON that the Javascript code receives from the Assistant service is on the right. Type any request to test your Chatbot. The system understands your intents and responds. You can see the details of how your input in the **user input** section was understood by examining the JSON data in the **Watson understands** section on the right side.

11. Test your local application

1. Test your application.

For example, if you type *where is the pool*, the JSON data shows that the system understood the *#hotel_location* intent with a high level of confidence, along with *@hotel_amenity* entity with the value *pool*.

The screenshot shows a chat interface on the left and a JSON output window on the right. The chat interface displays a conversation where the user asks "where is the pool?" and the system responds "The pool is on the second floor." The JSON output window, titled "Watson understands", shows the system's internal processing of the user's input. Red boxes highlight key parts of the JSON: the intent "hotel_locations" with a confidence of 1, the entity "hotel_amenity" with a location of [13, 17] and a value of "pool" with a confidence of 1, and the final response text "The pool is on the second floor."

Good Morning!
I am Watson. I can answer questions about the Hotel
How can I help you today?

where is the pool?

The pool is on the second floor.

```
Watson understands
1 {
2   "intents": [
3     {
4       "intent": "hotel_locations",
5       "confidence": 1
6     }
7   ],
8   "entities": [
9     {
10      "entity": "hotel_amenity",
11      "location": [
12        13,
13        17
14      ],
15      "value": "pool",
16      "confidence": 1
17    }
18  ],
19  "input": {
20    "text": "where is the pool?"
21  },
22  "output": {
23    "generic": [
24      {
25        "response_type": "text",
26        "text": "The pool is on the second floor."
27      }
28    ],
29    "text": [
30      "The pool is on the second floor."
31    ],
32    "nodes_visited": [
33      "node_8_1534432161729",
34      "node_1_1534492280438",
35      "node_2_1534492334068",
36      "node_14_1534435053398",
37      "node_15_1534435076277"
38    ]
39  }
40 }
```

2. In order to populate your data base, continue to test the application like you did in the **try it out** panel.

To prepare the improvement, you will enter intents and entities unknown by Watson

3. Type *I would like an apple pie.*

Good Afternoon!
I am Watson, nice to meet you
The forecast today in Nice is Partly cloudy.
Low 13C.

i would like an apple pie

What kind of restaurant or what food would you like?

```
Watson understands
1 {
2   "intents": [
3     {
4       "intent": "eat",
5       "confidence": 0.812211275100708
6     }
7   ],
8   "entities": [],
9   "input": {
10    "text": "i would like an apple pie"
11  },
12  "output": {
```

Watson identifies #eat intent but doesn't identify any entity. #eat is a wrong classification.

4. Type *French* and *now* to go out the Find restaurant branch.

5. Type *Where can I grab something to eat?*

Good Morning!
I am Watson. I can answer questions about the Hotel
How can I help you today?

Where can I grab something to eat?

I understand that you'd like to locate something in the hotel. Please ask me for the specific amenity you are looking for, (gym, restaurant, pool, sauna) or you may call the front desk for directions.

```
Watson understands
1 {
2   "intents": [
3     {
4       "intent": "hotel_locations",
5       "confidence": 0.6927498817443848
6     }
7   ],
8   "entities": [],
9   "input": {
10    "text": "Where can I grab something to eat?"
11  },
12  "output": {
13    "generic": [
14      {
15        "response_type": "text",
16        "text": "I understand that you'd like to locate som
17      }
18    ],
```

Hotel_location intent could be a right classification , but we want to associate this user's example to #eat intent.

6. Type *I want an Iphone*

Good Morning!

I am Watson. I can answer questions about the Hotel

How can I help you today?

I want an Iphone

I didn't understand. You can try rephrasing.

```
Watson understands {...}
1 {
2   "intents": [
3     {
4       "intent": "General_Jokes",
5       "confidence": 0.26708720326423646
6     }
7   ],
8   "entities": [],
9   "input": {
10    "text": "I want an Iphone"
  }
```

Watson identifies the intent *#General_Jokes*. This example is not relevant in our use case.

You can continue to test and submit input to Watson with sentences which make sense for you.

Sending utterances and generating logs with Watson Studio

This chapter proposes a way to populate automatically the log of your Watson assistant workspace by using Watson Studio. A notebook is provided to leverage the WA Python SDK.

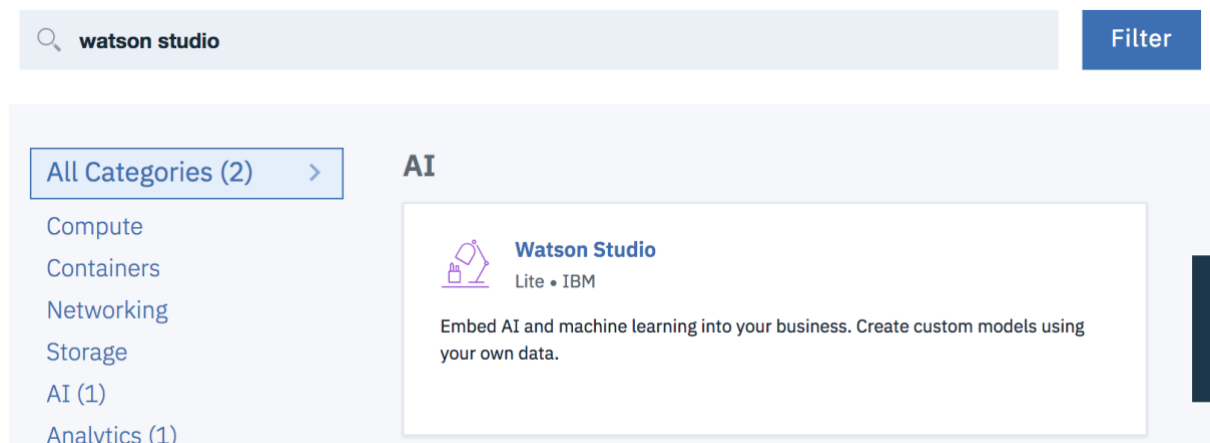
12. Create a Watson Studio instance

You can use also an existing instance of Watson Studio in the right region. Go to the next step in this case.

1. Go back on your IBM Cloud Dashboard.
2. Create a new resource (**create resource** button)

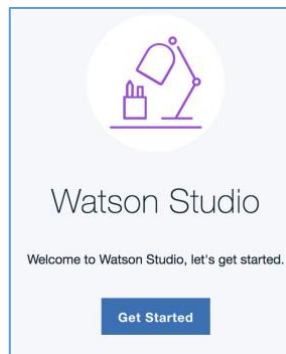
Note: be sure to create the Watson studio instance and the Watson Assistant in the same region.

3. Look for Watson studio



4. Then click on **Watson Studio** tile
5. Select *US south / Dallas* as region
6. Keep the default group and click **Create**

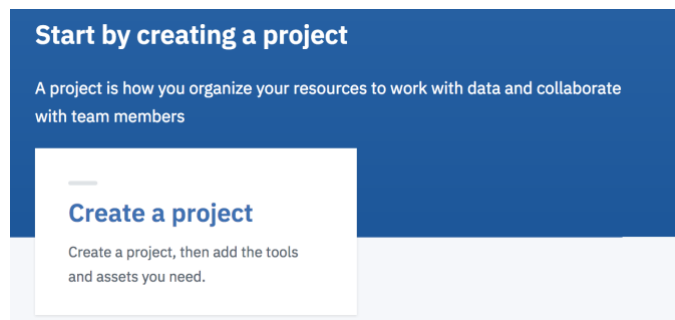
7. Click **Get Started**



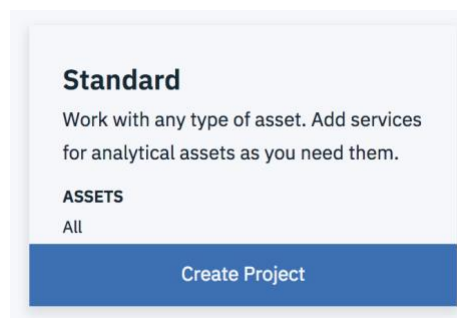
You are going to use Watson Studio to test the efficiency of your Ground Truth.

13. Create Watson Studio resources

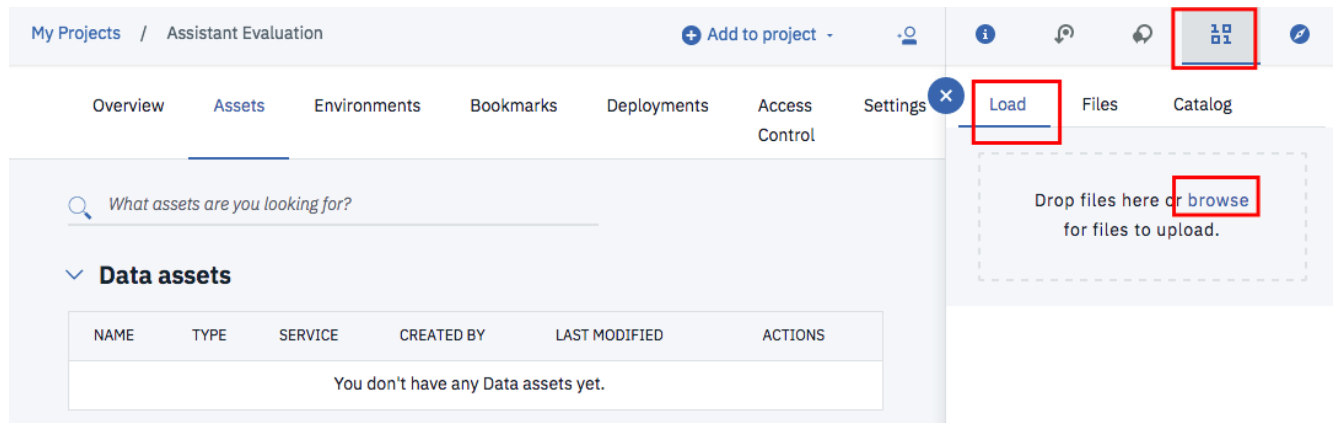
1. Click on new project tile, you can use an existing project (with Jupyter Notebooks) and just add a new notebook



2. Select the **Standard** tile and click **Create Project**.

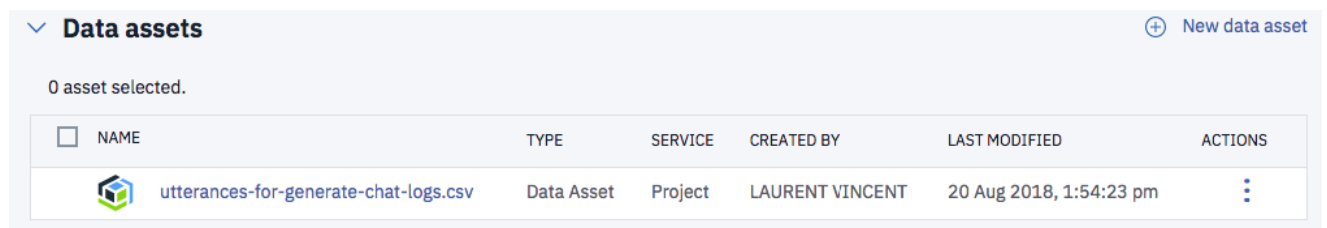


3. Name it *Assistant Evaluation*
4. Keep the default storage, click **Create**
5. Select **Assets** tab
6. Select **Load** tab from the **data** panel.

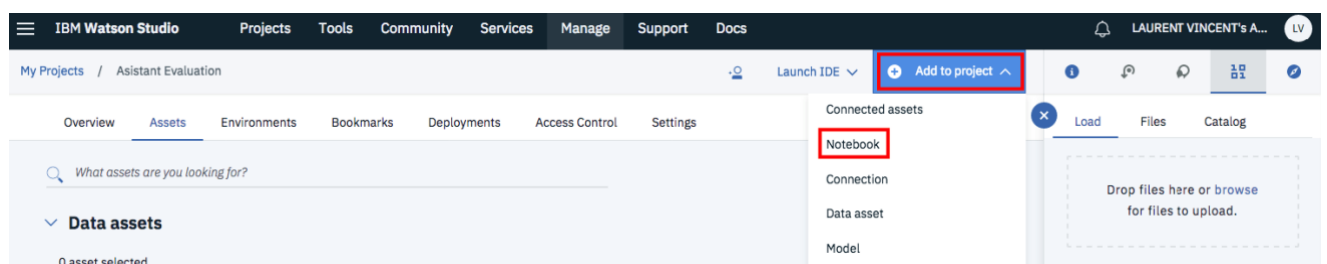


7. Upload *utterances-for-generate-chat-logs.csv* csv file created to populate your logs.

Then your data set is available in your project



8. Click **Add to project** and **Notebook**



9. Select **From File** tab, then **browse**

New notebook

Blank **From file** From URL

Name*

Type Notebook Name here

Description

Type your Description here

Notebook file*

Parcourir... Aucun fichier sélectionné.

10. Select the notebook *[Sending utterances and generating logs.ipynb](#)* provided in the Box

11. Select the right runtime *[Python 3.5 Free](#)*

Notebook file*

Parcourir... 01-Sending utterances and generating logs sample.ipynb

Import a Python, Scala, or R notebook file (.ipynb) from your local device.

Select runtime* Includes notebook environments ⓘ

Default Python 3.5 Free (1 vCPU and 4 GB RAM)

The selected runtime has 1 vCPU and 4 GB RAM and is free.
[Learn more](#) about capacity unit hours and Watson Studio pricing plans.

12. At the bottom, Click **Create Notebook**

Sending utterances and generating logs

This notebook demonstrates a technique to programmatically Send utterances and

Sometimes you need a quick way to simulate a user and call the /message API with a workspace and generate data for the Improve section of the Conversation UI. you can use a chat-logs CSV file to do this.

This notebook will demonstrate how the Watson Assistant API can be directly accessed from a user GUI.

This notebook runs on Python 3.5 So, on the top right be sure to run the right Kernel

Tips:

13. Now you can follow the instructions in the notebook

Once the notebook executed, the Improve page should look like this:



Improve Conversation

Now, You are going to work with 2 workspaces:

The one where you built your dialog and where you have populated the logs. It will be use to simulate the production environment

A new one , you will create, to apply your improvements which will simulate the test environment.

14. Navigate on Overview page

1. Click on the hamburger menu (top left) and click on **Improve**

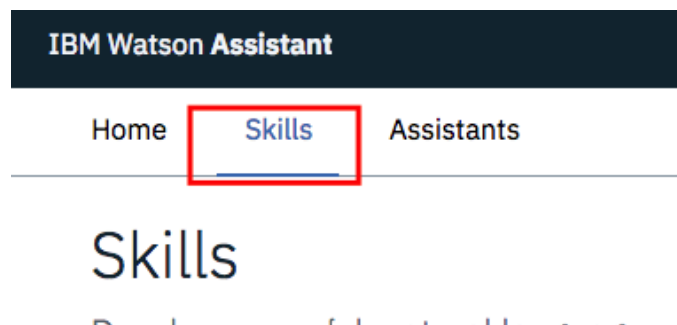
You should get some data. Be sure to select the right date.

2. Navigate to the page to work with the different features.

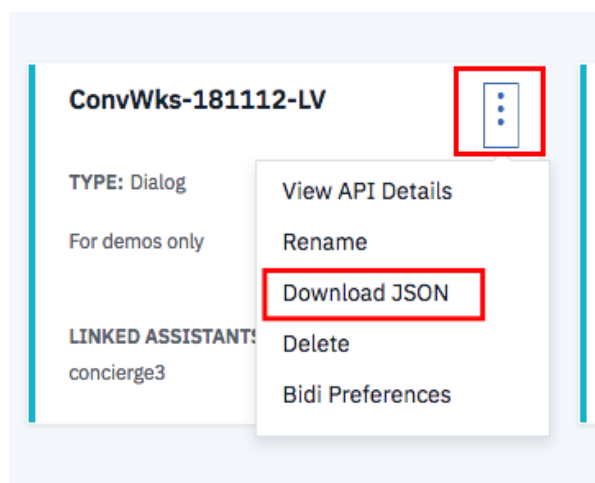
15. Create a test-experimental workspace

You are going to create a workspace to do the enhancements of your conversation as we cannot do this in production..

1. Go back to the **Skills** tabClick on workspace



2. On your skill tile, click on the contextual menu and **Download JSON** to save it on your local file system



3. Click **Create New** skill

4. Go to **Import skill** tab, click **choose JSON File**, then select the file previously saved on your local file system.
5. Click **Import**
6. Rename the new skill, as example *ConvWks-180401-LV Test* (use the tile menu to do it) that the skill not assigned to an assistant.

Now your new test skill is available.

16. Selecting data source / Deployment ID

As you added the deployment ID *ChatbotProduction* in your message API calls, you will be able to use utterances of your current skill from any other skills created in your Watson Assistant service or filter the data according to the channel, the environment or the user group.

1. Go to the Overview page of Improve feature of your *Test experimental workspace*.
2. Select **Data source** then click **Show deployment IDs**

The screenshot shows the 'Improve' feature interface for a skill named 'ConvWks-180820-LV-all-source Test / Improve'. The 'Data Source' dropdown is open, showing the current selection 'ConvWks-180820-LV-all-source' and a list of other skills including 'Concierge' and '60a69a81-4ea5-420f-bfe3-adc53b9ed...'. A 'Show deployment IDs' button is highlighted in red. The main content area shows a 'Filter' section with 'Intents' and 'Entities' dropdowns, and a summary table with columns for 'Total conversations', 'Avg. msg. per conversation', and 'Max. conversations', all showing a value of 0.

Now you have the possibility to select *ChabotProduction* data source.

3. Select *ChatbotProduction* data source.

Note: When switching to another data source, the Conversation service checks utterances for an element called Deployment ID. Deployment IDs are unique identifiers in the Conversation service API that you add to your message API calls.

Note: While **Data source:** now shows the source of the utterances you are using to improve this skill, the top of the page still shows the workspace you are applying changes to.

17. Adding an entity value or synonym


1. At the bottom, on Top intents Select the *#eat* intent.

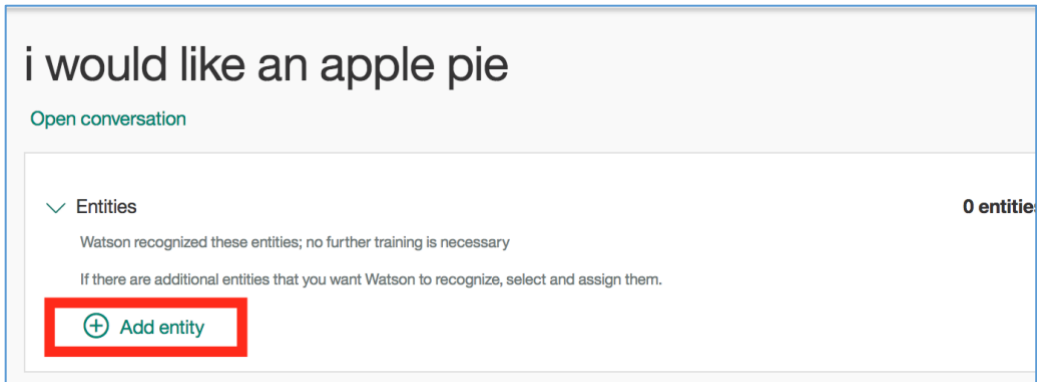
Intent	Total
#eat	7
#local_recommend	6
#exit	4
#hotel_hours	4

Watson opens the User conversations page and returns all records using *#eat* intent.

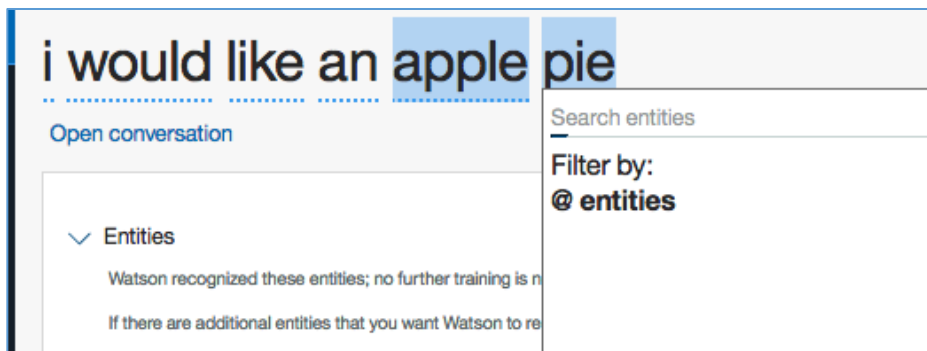
The screenshot shows the Watson User Conversations interface. At the top, there is a filter for '#eat' and a search bar. Below the filter, there are two dropdown menus for 'Intents' and 'Entities'. The main content area shows a list of user statements. The first statement is 'i would like an apple pie' with the intent '#eat' and 'No entities' identified. The second statement is 'I want an Iphone' with the intent '#eat' and 'No entities' identified.

You find that Watson didn't identify *Apple pie* as entity. *Apple pie* could be added as synonym of one of your entity to improve your chatbot.

2. To add an entity value or synonym, select the  edit entities icon on the left of the selected record.
3. Click **Add entity**

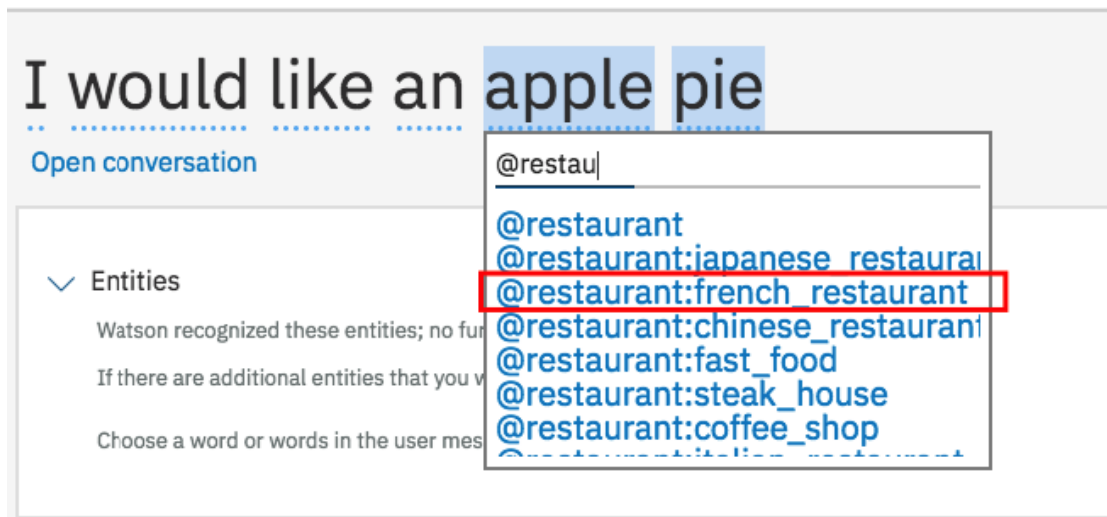


4. Press shift and select the words **apple** and **pie**.



You must enter an entity to which the highlighted phrase will be added as a value. Begin typing in the entry field, the list of entities will be populated. To add the highlighted phrase as a synonym for an existing value, enter the entity:value.

5. Right now, enter restaurant and select [@restaurant:French_restaurant](#) entity



then you will get this:

I would like an apple pie

Open conversation

Entities

Watson recognized these entities; no further training is necessary

If there are additional entities that you want Watson to recognize, select and assign them.

apple pie = @restaurant:french_restaurant

Choose a word or words in the user message to add as an entity, value, or synonym.

6. Click on **Save** button

if you look at the entity @restaurant the synonym apple pie has been added to french.

@restaurant								
+ Add a new synonym value		+ Add a new pattern value						
<input type="checkbox"/>	french_restaurant	french restaurant	brasserie	frogs legs	macaron	french food	french	apple pie
<input type="checkbox"/>	pizza_restaurant	pizza restaurant	pizza	pizza pie	pizzeria			
<input type="checkbox"/>	chinese_restaurant	chinese restaurant	spring roll	noodles	chinese food	chinese		
<input type="checkbox"/>	fast_food	fast food	hamburger	junk food				
<input type="checkbox"/>	steak_house	steak house	beef	meat	tex-mex	texan	mexican	
<input type="checkbox"/>	coffee_shop	coffee place	coffee shop	dunkin	dunking	starbucks		
<input type="checkbox"/>	italian_restaurant	italian cuisine	italian food	italian place	italian restaurant	pasta	italian	
<input type="checkbox"/>	japanese_restaurant	japanese food	japanese restaurant	sushi	japanese			

18. Correcting an intent

1. Click on **User Conversations** tab

2. As Search user statements, enter *where can I grab*

Overview **User conversations** Data source: ChatbotProduction

Filter Refresh data Last updated: 2:34 PM **This week** 20 Aug 18 to 20 Aug 18 by the **hour**


Intents Entities

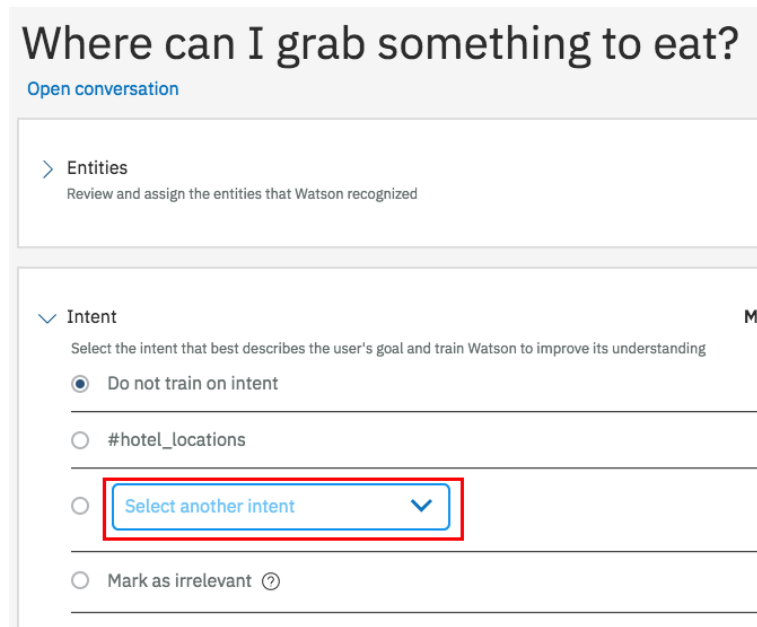
Showing 1 through 50 of 156 results Newest first

3. Look for the sentence, *where can I grab something to eat*

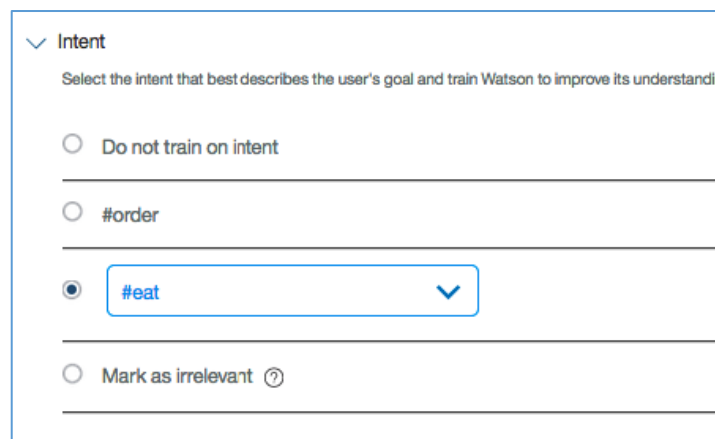
4. You can see that Watson identifies the intent `#hotel_locations` and we want to assign it to the intent `#eat`



5. To update an intent value, select the  edit icon on the left of the selected record.



6. From the drop-down list of intent, select `#eat` intent.



7. Click on **Save** button
8. You can repeat the previous steps for the user's example *Where can I grab a hamburger ?* and move it from `#local_recommend` to `#eat` intent.

if you look at the intent #eat, the sentence has been added as utterance.

Intent name
#eat

Description
Add a description to this intent




Add user examples
Add user examples to this intent

[Add example](#)

User examples (10) ▼

- Hello I'm starving I'm looking for a place to eat ✎
- I am hungry ✎
- I am looking for a restaurant ✎
- I am really hungry you know a good place to eat around here ✎
- I am starving ✎
- I want pasta ✎
- I want some food ✎
- I want to eat at an Italian restaurant. What are the best ones around? ✎
- I would like to eat something ✎
- Where can I grab something to eat? ✎

Note: you can review the whole conversation related to the selected record by using **open conversation**.

 Where can I grab something to eat?  #eat
Trained: 08/20/2018 @ 2:42 PM  No entities

08/20/2018 @ 11:42 AM [Open conversation](#)

19. Irrelevant input

You can "Mark as irrelevant" to indicate that the input is not related to your application.

Inputs marked as irrelevant are stored in the workspace and are included as part of the training data. **They cannot be accessed or changed later in the tooling.**

Note If you already have an intent for inputs that are out of scope or off topic, such as #off_topic, you should delete that intent and test your workspace with the use of "Irrelevant".

You are allowed 25,000 irrelevant examples per service instance.

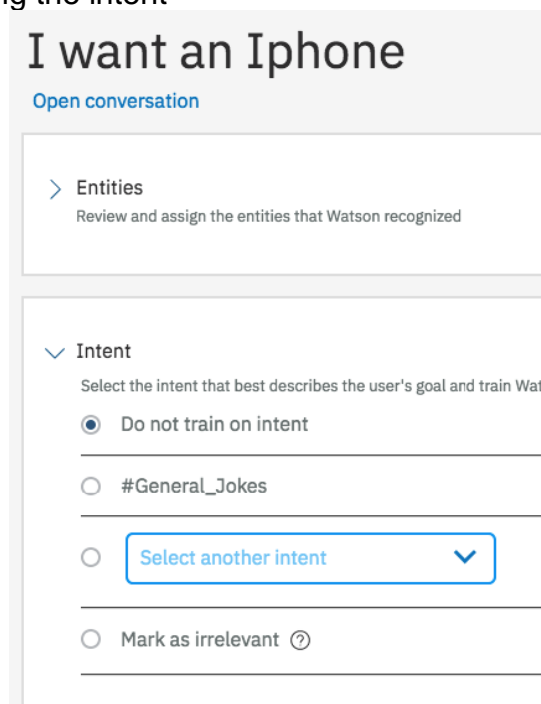
You have seen that Watson identifies *I want an Iphone* as #General_Jokes intent.



To eliminate all ambiguity, you have some options :

- you must create new intent such as #buy, when a user wants to buy something.
- Do not train on intent, which will not save this utterance as an example for training
- Mark this sentence as irrelevant

It can be done by editing the intent



20. Test your conversation updates

1. Return to your local application <http://localhost:3000>

2. Successively enter:

I want an apple pie

I want an Iphone

where can I grab something to eat

The virtual assistant should have the expected behaviors.

Test your improvements with Watson Studio

You are going to use Watson Studio to test the efficiency of your improvements by comparing the behavior of your prod workspace and test workspace. Go back to your Watson Studio instance

21. Prerequisite : create a cloudant service

One of the requirement of the next notebook is to create a Cloudant service

1. Go back to your IBM Cloud Catalog and create a cloudant instance or use a existing one.


Catalog

cloudant

All Categories (4) >

- Compute
- Containers
- Networking
- Storage
- AI
- Analytics

Databases

**Cloudant**
Lite • IBM
A scalable JSON document database for web, mobile, IoT, and serverless applications.

← View all

Cloudant

Lite • IBM

IBM Cloudant is a fully managed JSON document database. Cloudant is compatible with Apache CouchDB and accessible through a simple to use HTTPS API for web, mobile, and IoT applications. Cloudant is SOC2 and ISO 27001 compliant with HIPAA readiness optional for Dedicated Hardware environments. Cloudant Standard plan instances come with a 99.95% SLA. All data is encrypted at rest and over the wire. Cloudant JSON documents are stored in triplicate across three separate availability zones for in-region HA/DR in regions that support AZ's. Any Cloudant instance deployed from the Germany region/location will be in EU supported

Service name:
Cloudant-Watson-Studio-for-Assistant

Choose a region/location to deploy in: United Kingdom
Select a resource group: default

Available authentication methods:
Select a value

Legacy credentials enable login to Cloudant using HTTP Basic authentication

2. Once created, Go to **service credentials** page (upper right menu)
3. Expand **view credentials**
4. Copy url , username, password

Manage

- Service credentials**
- Plan
- Connections

Cloudant-Watson-Studio-for-Assistant

Resource Group: default Location: London

Service credentials

Credentials are provided in JSON format. The JSON snippet lists credentials, such as the API key and secret, as well as connection information for the service. [Learn more](#)

Service credentials New credential +

Items per page 10 | 1-1 of 1 items 1 of 1 pages < 1 ▾

<input type="checkbox"/> KEY NAME	DATE CREATED	ACTIONS
<input type="checkbox"/> Service credentials-1	AUG 20, 2018 - 02:09:50 PM	View credentials ▾ 🗑️

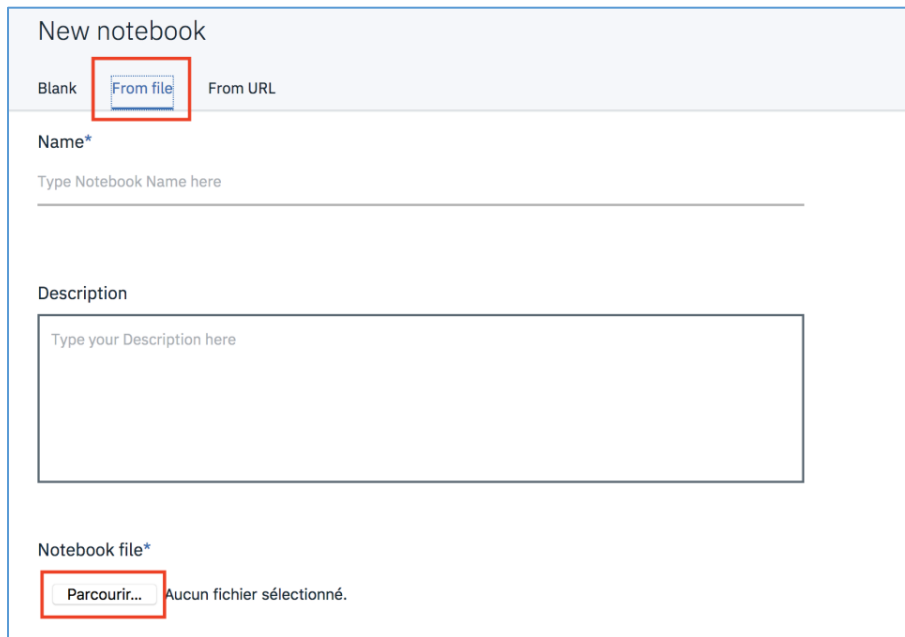
```

{
  "apikey": "0110[REDACTED]000",
  "host": "6a852[REDACTED]cloudant.com",
  "iam_apikey_description": "Auto generated apikey during resource-key operation for Instance - crn:v1:bluemix:public:cloudantnosqldb:eu-g
b:a/d8a3425072408a6a10ec4e2[REDACTED]10195664266:",
  "iam_apikey_name": "apikey-611916ad7bb9",
  "iam_role_crn": "crn:v1:bluemix:public:iamsts:eu-gb:3-900000000000:4335-8f0
0-e279b4cc77a3",
  "password": "6ca8a6f0ff0[REDACTED]08470ca4",
  "port": 443,
  "url": "https://6[REDACTED]000cca9c19c00a12e20ce900a070ca406a45261e-5",
  "username": "6a462e1[REDACTED]000"
}

```

22. Create Watson Studio resources

5. Go back to your Watson studio page, Open the *Assistant Evaluation* project
6. Select **Assets** tab
7. Click on **New Notebook**
8. Select **From File** tab, then **browse**



New notebook

Blank **From file** From URL

Name*

Type Notebook Name here

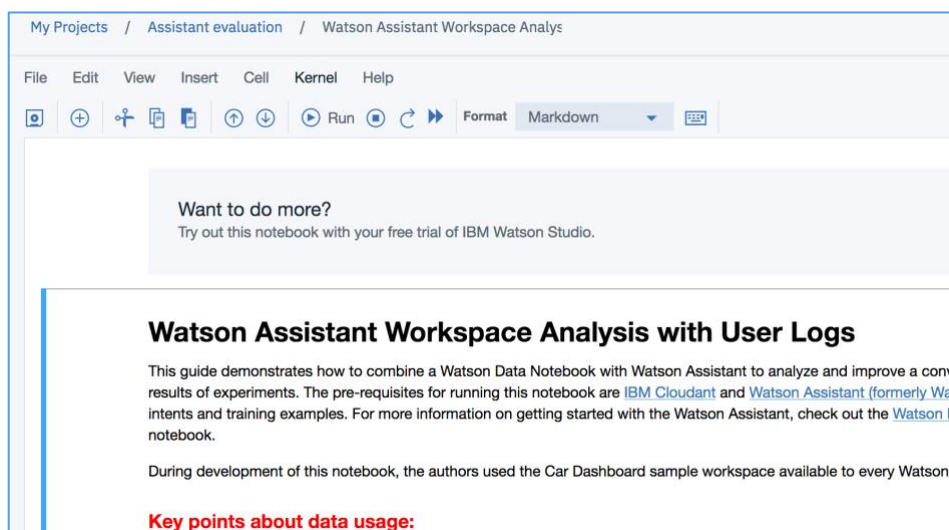
Description

Type your Description here

Notebook file*

Parcourir... Aucun fichier sélectionné.

9. Select the notebook *Watson Assistant Workspace Analysis with User Logs.ipynb* provided in the Box
10. Select the right runtime *Python 3.5 Free*
11. Click **Create Notebook**



My Projects / Assistant evaluation / Watson Assistant Workspace Analysis

File Edit View Insert Cell Kernel Help

Run

Want to do more?
Try out this notebook with your free trial of IBM Watson Studio.

Watson Assistant Workspace Analysis with User Logs

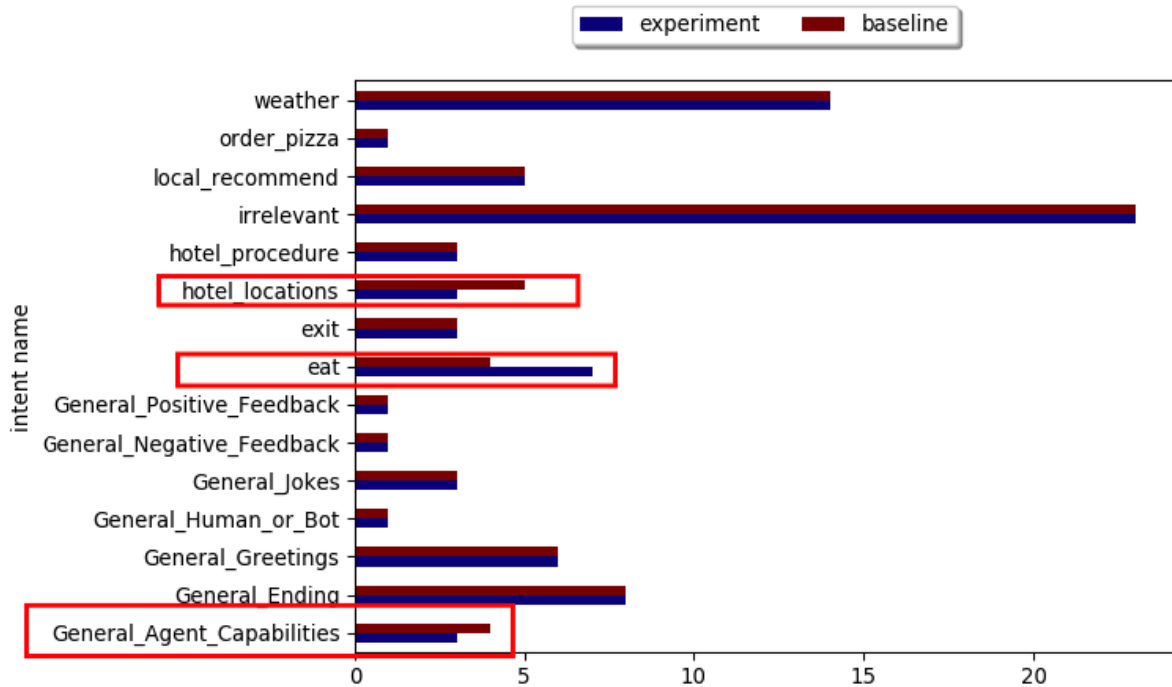
This guide demonstrates how to combine a Watson Data Notebook with Watson Assistant to analyze and improve a conversational interface. The pre-requisites for running this notebook are [IBM Cloudant](#) and [Watson Assistant \(formerly Watson Conversation\)](#) intents and training examples. For more information on getting started with the Watson Assistant, check out the [Watson Assistant Getting Started](#) notebook.

During development of this notebook, the authors used the Car Dashboard sample workspace available to every Watson Assistant user.

Key points about data usage:

12. Now you can follow the instructions in the notebook

At the end of the execution, you should get the following graph.



You retrieve a delta for the intents *hotel_location* and *eat*. these are what you did
So it is expected.

You can see, also you update for the intent *General_Agent_Capabilities*. it means
that your update has some unexpected impacts which required investigation.

You can review the details in the last table:

Showing 82 of 82 rows

	conf_delta	intent_1	intent_2	intent_changed	text
3640014653	-0.01796796321868893	weather	weather	False	turn on the headlights?
	0.0	weather	weather	False	is it raining?
511276246	0.15733978748321537	General_Agent_Capabilities	eat	True	play something
2388813416	-0.0035366156646119418	General_Greetings	General_Greetings	False	k pm
121258544	-0.2423959732055665	eat	eat	False	i would like an apple

It is up to you to determine if this is an improvement or not.