

Build, Train, Save, Deploy and Test a Convolutional Neural Network Model using MNIST

This lab will use the [MNIST](#) computer vision data set to train a deep learning model to recognize handwritten digits. A single layer convolutional neural network will be built in the Watson Studio neural network designer, and then trained using the Watson Studio Experiment Builder. The trained model will be saved in the model repository, deployed, and then tested with sample image data. The lab consists of the following steps:

1. Set up the data files in IBM Cloud Storage.
2. Design the neural network
3. Train the model
4. Monitor the training progress and results
5. Save and Deploy the Trained Model
6. Test the Deployment

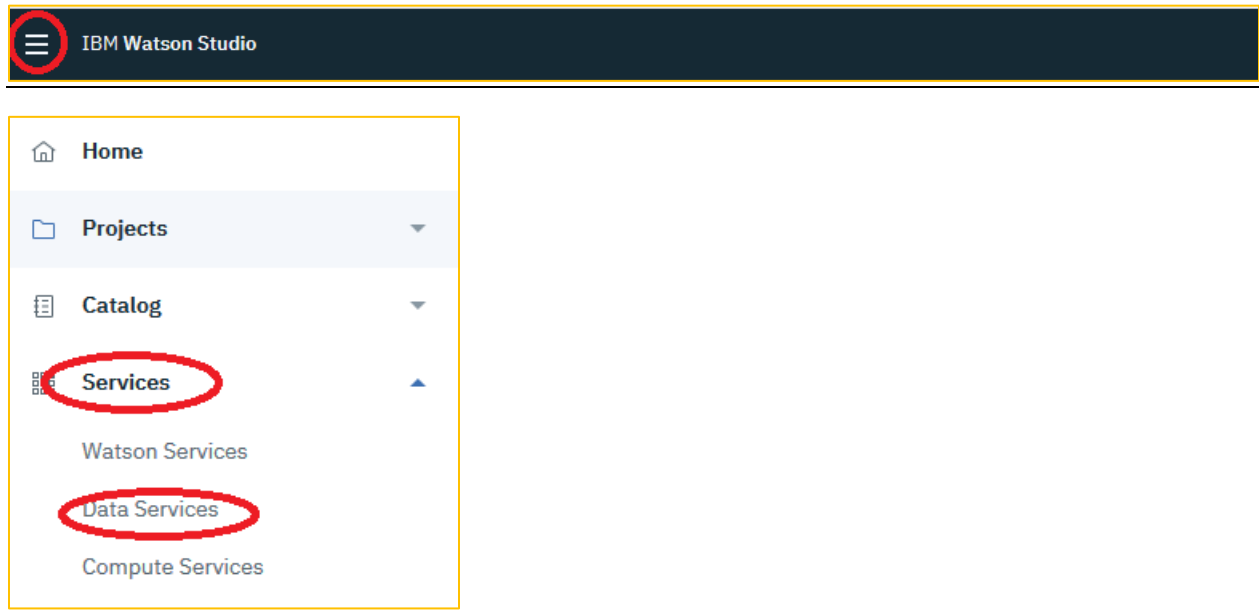
Step 1: Set up the Data Files in IBM Cloud Storage

Training a deep learning model using Watson Machine Learning relies on using Cloud Object Storage for reading input (such as training data) as well as for storing results (such as log files.)

1. Click [here](#) to download the mnist.zip file. Extract the files. Four files should be extracted:
 1. a training file (mnist-tf-train.pkl) – pickle format.
 2. a test file (mnist-tf-test.pkl)
 3. a validation file (mnist-tf-valid.pkl)
 4. test.json (will be used to test the deployed model)



- Return to Watson Studio, click on the  icon, then click on **Services**, and then **Data Services**.



- Select the vertical **ellipse** on the right-hand side of the cloud object storage entry, and then click on **Manage in IBM Cloud**.






- A new browser tab **Create Object Storage – IBM Cloud** is created. This is the IBM Cloud user interface to the object storage subsystem



5. Click on **Create**.

Resource list /

 cloud-object-storage-ns [View docs](#) [Aspera transfers](#) 

Resource group: Default Location: Global Add tags 

Buckets



🔍 Prefix filter ⓘ


Create +

Name	Public Access ⓘ	Location ⓘ	Storage Class	Created	Advanced ⓘ
------	-----------------	------------	---------------	---------	------------

6. Click on **Custom bucket**.

Resource list /

 cloud-object-storage-ns [View docs](#) [Aspera transfers](#) 

Resource group: Default Location: Global Add tags 

Create bucket

Get started by creating a bucket to store unstructured data. Create a custom bucket of your own or choose from our pre-defined configurations

Customize your bucket

Custom bucket

Create a bucket by selecting bucket configurations that meet your object storage needs.

Predefined buckets

Standard

Create a standard storage class bucket in a region close to you and a service credential to connect your application

Archive your data

Create a standard storage class bucket in a region close to you with an archive policy and a service credential to connect your application

7. Enter a unique name for the bucket - mnist-lab-train-xxx (replace xxx with your initials), click on **Cross-Region** for the **Resiliency**, and click on **us-geo** for the **Location**. **MAKE SURE YOU CHANGE THE LOCATION TO US-GEO BECAUSE IT DEFAULTS TO AP-GEO**. Scroll down and click on **Create bucket**.

Create bucket

Unique bucket name

mnist-lab-train-bzb

i **Bucket naming rules:**

- Must be unique across the **whole** IBM Cloud Object Storage system
- Do not use any personal information (any part of a name, address, financial or security accounts or SSN)
- Must start and end in alphanumeric characters (3 to 63)
- Characters allowed: lowercase, numbers and non-consecutive dots and hyphens

Resiliency

Cross Region

Location

us-geo

8. Scroll down and click **Create bucket**.

cloud-object-storage-ns

View docsAspera transfers

Resource group: DefaultLocation: GlobalAdd tags

For less active workloads that require infrequent data access (accessed once a month or less).

For cold workloads where data is primarily archived (accessed a few times a year).

Additional configuration (optional)

☐ Add Archive rule

The feature is currently not supported in the location you have selected. [Learn more](#)

☐ Add Expiration rules

☐ Add Retention policy

This feature is available for our Standard plan customers only. [See pricing](#)

Key Management Services (optional)

Services can only be added at bucket creation; additionally if the key is deleted later all bucket data will become inaccessible.

☐ Add Key Protect key

☐ Add Hyper Protect Crypto Services key

The feature is currently not supported in the location you have selected. [Learn more](#)

Additional Services (optional)

☐ IBM Cloud Activity Tracker with LogDNA (Third Party)

☐ IBM Cloud Monitoring with Sysdig (Third Party)

Create bucket

9. Navigate to the directory where the 3 mnist files are stored. **Select these 3 files and drag and drop where indicated.**

Objects

Search

☐

Object Name	Size	Last Modified
-------------	------	---------------

Drag and drop files or folders to upload them.

Computer > Local Disk (C:) > Presentations > Machine Learning > 9-6 > Lab-3 > data

OpenBurnNew folder

Name	Date modified	Type	Size
mnist.zip	6/5/2018 10:38 AM	Compressed (zipp...	11,350 KB
mnist-tf-test.pkl	9/1/2018 11:59 AM	PKL File	7,667 KB
mnist-tf-train.pkl	9/1/2018 11:59 AM	PKL File	38,331 KB
mnist-tf-valid.pkl	9/1/2018 11:59 AM	PKL File	7,667 KB

10. Click on **Buckets** to add a second bucket.

Resource list / cloud-object-storage-ns /
mnist-lab-train-bzb

View docs Aspera transfers

Objects

Prefix filter

Upload

Object name	Size	Last modified
mnist-tf-test.pkl	7.5 MB	03/25/2020 3:56:52 PM
mnist-tf-train.pkl	37.4 MB	03/25/2020 3:57:10 PM
mnist-tf-valid.pkl	7.5 MB	03/25/2020 3:57:14 PM

Items per page: 10 1-10 of items

page 1

FEEDBACK

11. Click on **Create**.

Resource list /
cloud-object-storage-ns

View docs Aspera transfers

Resource group: Default Location: Global Add tags

Buckets

Prefix filter

Create

Name	Public Access	Location	Storage Class	Created	Advanced
------	---------------	----------	---------------	---------	----------

12. Name it mnist-lab-results-xxx, where xxx are your initials. Follow the procedure above to create the second bucket. No files need to be added. **MAKE SURE YOU CHANGE THE LOCATION TO US-GEO.**

Create bucket

Unique bucket name

mnist-lab-results-bzb

i **Bucket naming rules:**

- Must be unique across the **whole** IBM Cloud Object Storage system
- Do not use any personal information (any part of a name, address, financial or security accounts or SSN)
- Must start and end in alphanumeric characters (3 to 63)
- Characters allowed: lowercase, numbers and non-consecutive dots and hyphens

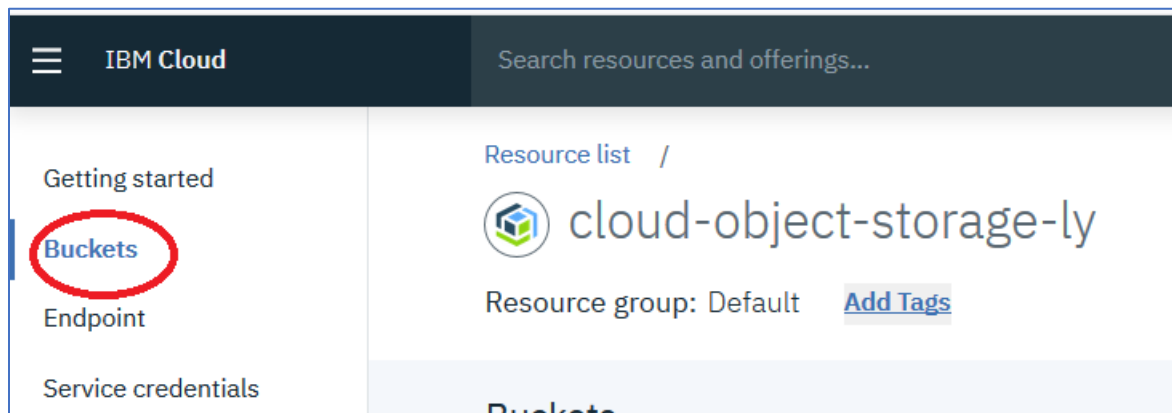
Resiliency

Cross Region

Location

us-geo

13. Click on Buckets.

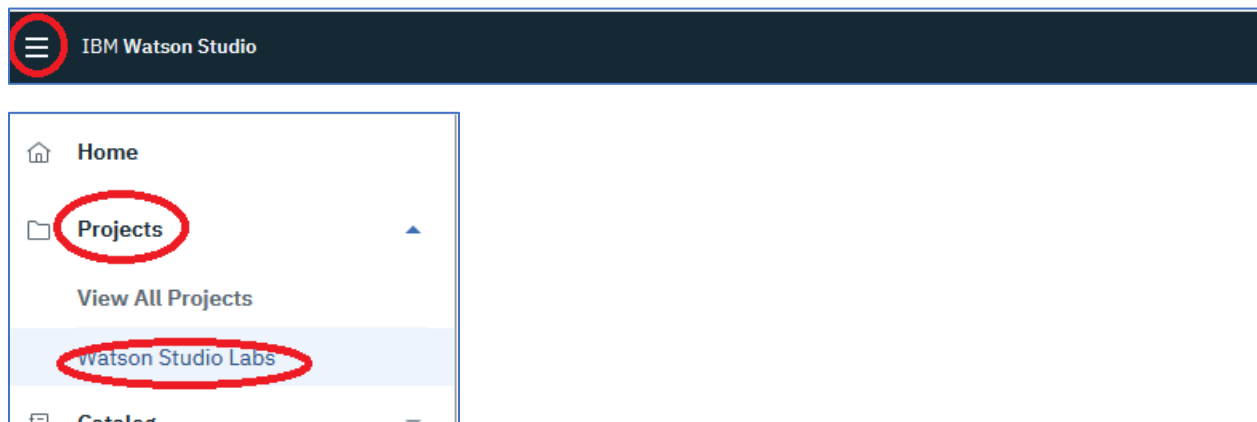


14. The Cloud Object Storage panel should appear as below. Note, you may not have the “htproject” bucket listed.

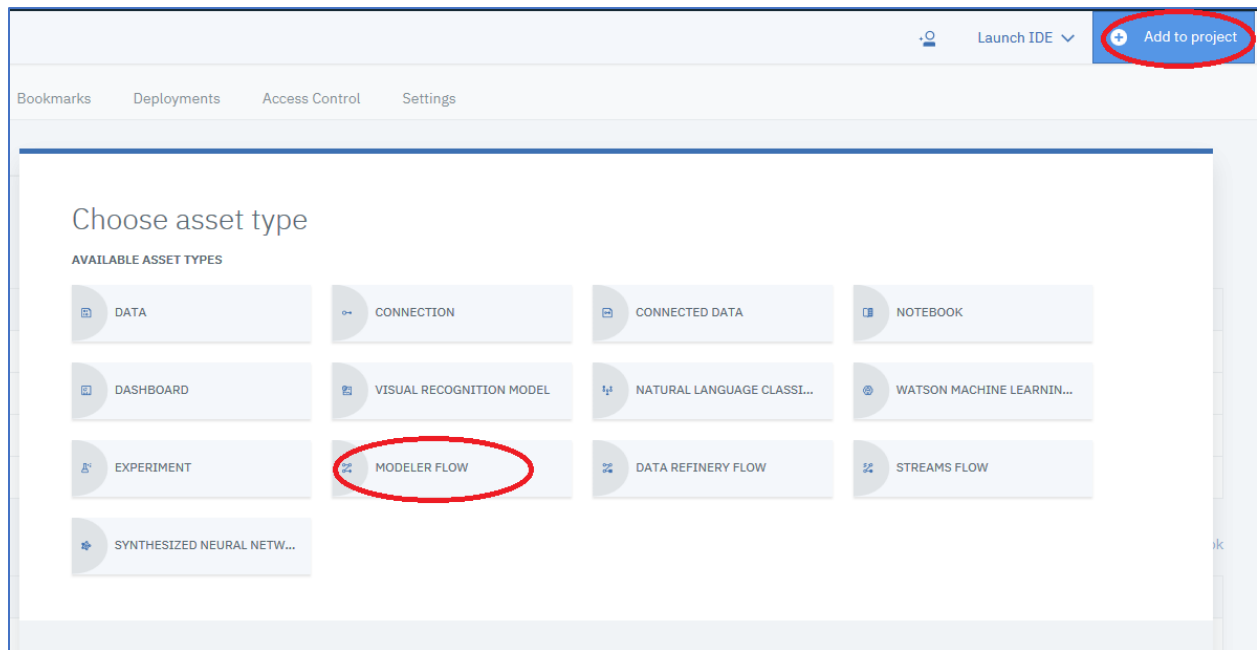
Buckets						
Prefix filter ⓘ			Create +			
Name	Public Access ⓘ	Location ⓘ	Storage Class	Created	Advanced ⓘ	
htproject-donotdelete-pr-ajgmcsjehgzeyb		us-geo	Standard	03/03/2020 12:29:38 PM	View	⋮
mnist-lab-results-bzb		us-geo	Standard	03/25/2020 5:17:08 PM	View	⋮
mnist-lab-train-bzb		us-geo	Standard	03/25/2020 3:56:08 PM	View	⋮

Step 2: Design the Neural Network and Publish Training Definition

1. Return to Watson Studio, and click on the  icon, then click on **Projects**, then **Watson Studio Labs**.



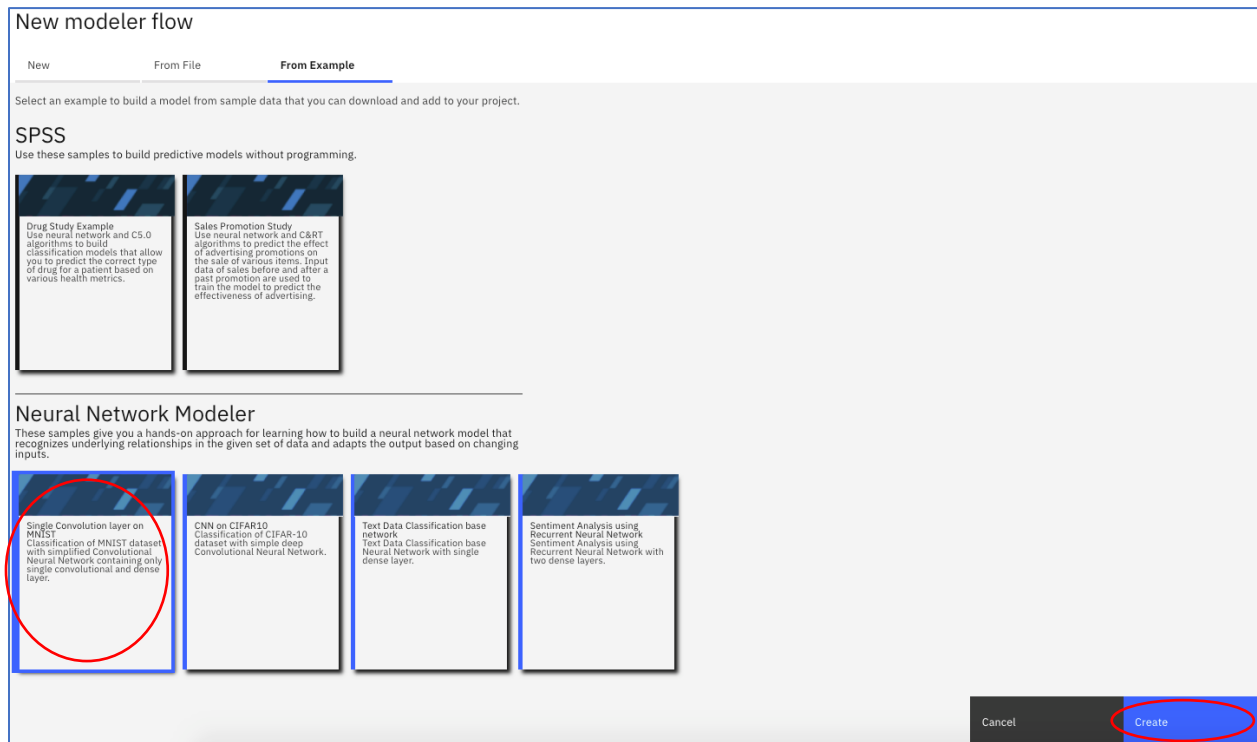
2. Click on the **Add to project** and then click on **MODELER FLOW**.



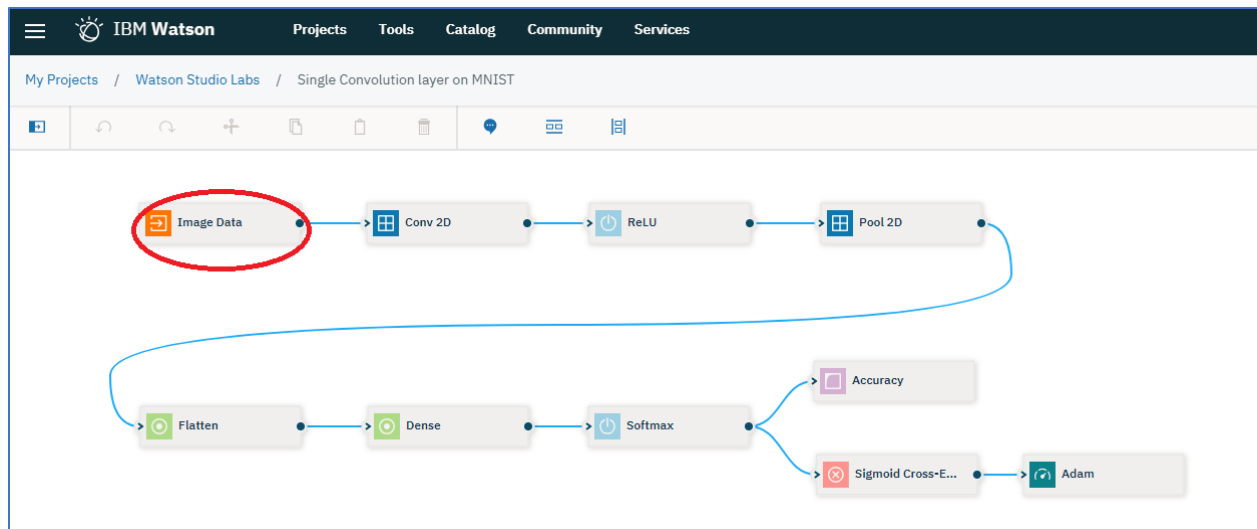
3. Click on **From example**.

The screenshot shows the IBM Watson Studio 'Modeler' page. The 'From example' tab is selected and circled in red. Below the tabs are fields for 'Name*' (with a 50-character limit) and 'Description' (with a 500-character limit). At the bottom, there are radio buttons for 'Select flow type' (Modeler Flow selected, Neural Network Modeler BETA) and 'Runtime' (IBM SPSS Modeler selected, Scala Spark 2.1 BETA).

4. Click on the **Single Convolution Layer on MNIST** and then click on **Create**



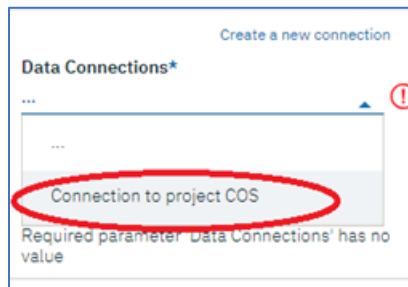
5. A standard convolution neural network (CNN) architecture is displayed on the Neural Network Modeler canvas. The Neural Network is represented in a graphical form instead of code. You will find a sidebar on the left of the screen containing all available neural network components. The Neural Network Modeler enables the user to drag and drop nodes representing the different layers of a Neural Network and to connect them to create a **Flow**. Here, we have an already created neural network flow via the example.
6. We still need to provide our neural network with a way to access the data we uploaded earlier. We configure the **Image Data** node for this purpose. Double-click on the **Image Data** node.



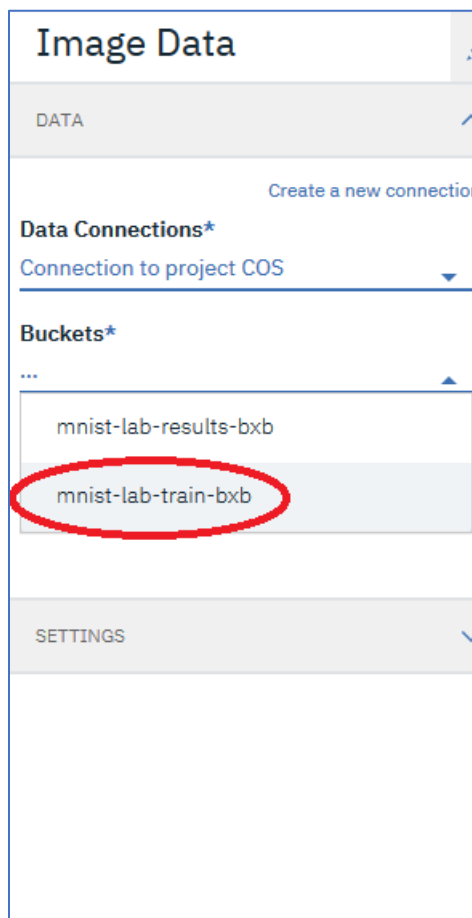
7. Leave the default **Connection Name** and click **Create Connection**.

The screenshot shows the "Image Data" configuration panel. The "DATA" section is expanded, showing a message: "There are no data connections in this project. Would you like to connect your default Cloud Object Storage instance to your project?". Below this, the "Connection Name" field is set to "Connection to project COS". The "Create a connection" button is circled in red.

8. Click on the downward triangle icon ▼ underneath **Data Connections***. Click on the connection that was just created.



9. Click on the downward triangle icon ▼ underneath **Buckets***, and then click on the **mnist-lab-train-xxx** where “xxx” are your initials.



10. Click on the ▼ icon underneath **Train data file*** and select the **mnist-tf-train-pkl**. Assign the **Test data file** (**mnist-tf-test-pkl**), and **Validation data files** (**mnist-tf-valid-pkl**) in the same way and then click on **Close**.

Select files from your bucket to specify train, validation and test data.



Train data file*
mnist-tf-train.pkl

Test data file
mnist-tf-test.pkl

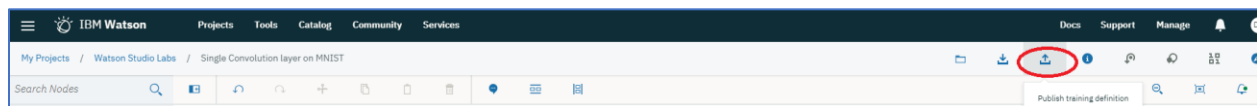
Validation data file
mnist-tf-valid.pkl

Close

11. Explore the neural network flow modeler options

1. Click on the  icon to see the list of neural network component categories that are available
2. Explore the contents in each category. Hover over the components to get a pop-up description.
3. Click on the download icon  to see the multiple options for code generation.

12. Click on the **Publish** icon to create a training definition.



13. Enter a name for the training definition (or leave the default) and select the Machine Learning service that you created. Note, it will not be named Machine Learning unless that is the name that you used. Click on **Publish**.

Publish Training Definition
Fill name and description then select WML instance to publish.

Name*
Single Convolution layer on MNIST


Description
Optional Description

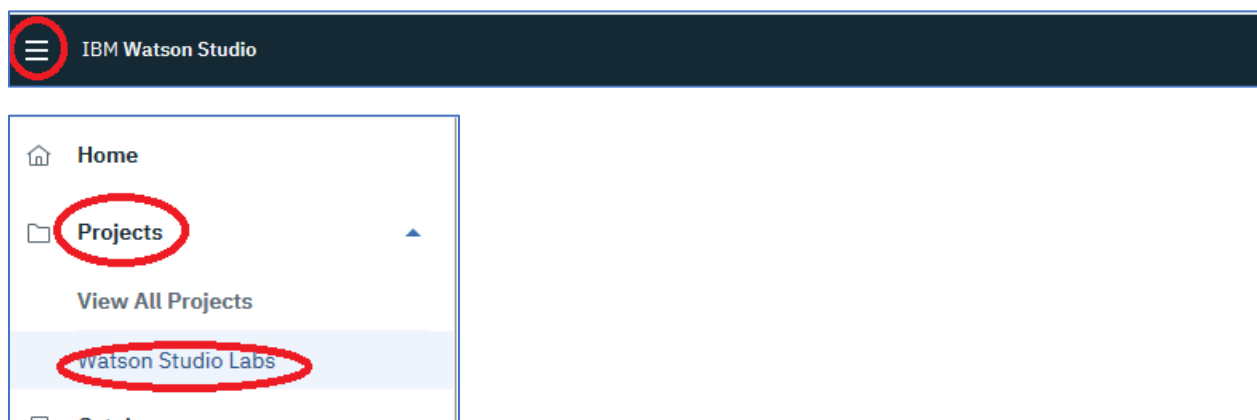
Select WML Instance*
Machine Learning

Cancel Publish

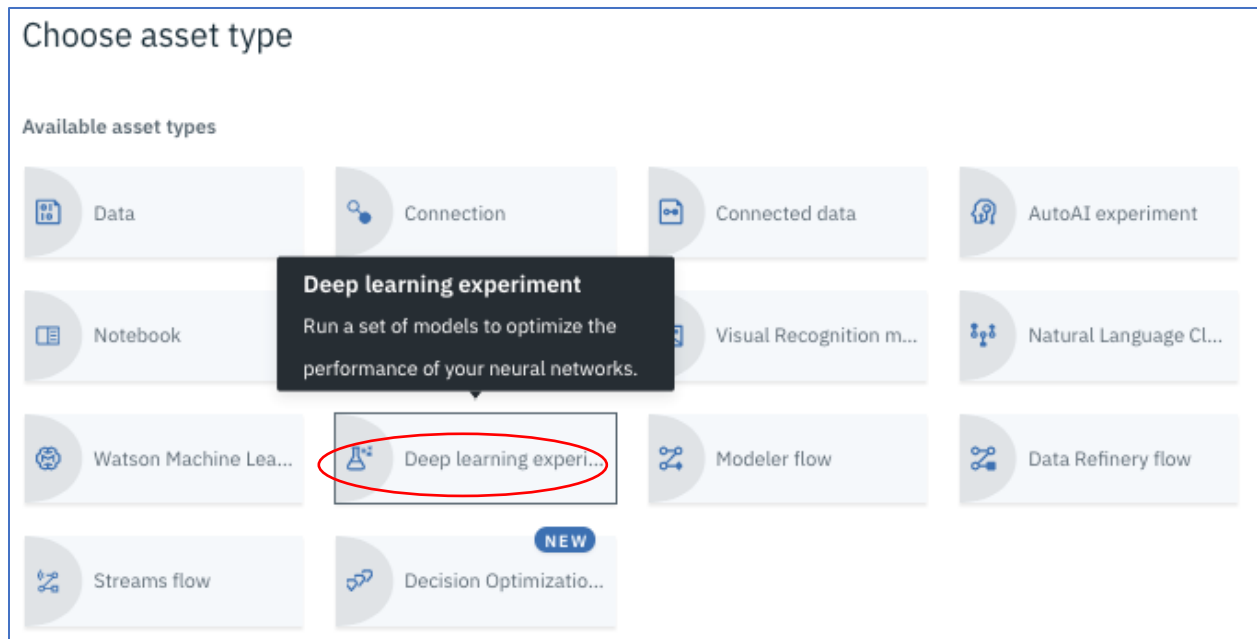
Step 3: Train the Model using Experiment Builder

As part of the model building process, we want to be able to compare different algorithms, and/or different algorithmic parameters to determine the best model. Experiment Builder is a facility in Watson Studio that supports this effort. Different training runs can be defined and run in parallel and their results can then be compared. In this lab, we will define only one training run to minimize the training time.

1. Return to the Watson Studio Labs **Assets** panel by clicking on the  icon, then click on **Projects**, and then **Watson Studio Labs**. Click on the **Assets** tab if the Assets panel is not displayed.



2. Click on **Add to project**, and then click **Deep learning experiment** to create a new Experiment.



3. Enter an Experiment **Name**, select the **Machine Learning** service, and then click on **Select** to assign a Cloud Storage bucket.

Define deep learning experiment details

Name

Single Convolution Layer on MNIST

Description

Experiment description

Machine Learning Service

Machine Learning Service

WatsonMachineLearning

Cloud Object Storage bucket for storing training data files

Select

Cloud Object Storage bucket for storing results

Associate training definitions

Add training definition +

Name	Compute plan
No training definitions associated.	

☐ Use global execution command (override training definition values)

Cancel Create and run

4. Select **Existing connections**, and then select the **Connection to project COS** connection.

The screenshot shows the 'Existing' tab selected in a configuration window. Below the tab, there is a section titled 'Cloud Object Storage connection'. A dropdown menu is open, showing the option 'Connection to project COS' which is circled in red.

5. We now need to assign the Training buckets. Select **Existing** underneath **Bucket containing training source data** and click on mnist-lab-train-xxx, where “xxx” are your initials. Click on **Select**.

The screenshot shows the 'Existing' tab selected in a configuration window. Below the tab, there is a section titled 'Bucket containing training source data'. Under this section, the 'Existing' radio button is selected and circled in red. Below the radio buttons, a dropdown menu is open, showing the option 'mnist-lab-train-bzb' which is circled in red. At the bottom right of the window, there are two buttons: 'Cancel' and 'Select', with the 'Select' button circled in red.

6. Click on **Select** underneath **Cloud Object Storage** for storing results.

Define deep learning experiment details

Name

Single Convolution Layer on MNIST

Description

Experiment description

Machine Learning Service

Machine Learning Service

WatsonMachineLearning

Cloud Object Storage
bucket for storing training data files

Source: [Connection to project COS / mnist-lab-train-bzb](#) [Update](#)

Cloud Object Storage
bucket for storing results

[Select](#)

- Follow the same procedure used to assign the training bucket to assign the results bucket. Assign bucket mnist-lab-results-xxx, where “xxx” are your initials, and then click on **Select**.

Existing New

Cloud Object Storage connection

Connection to project COS

Bucket containing results data

☒ Existing ☐ New

mnist-lab-results-bzb

Cancel [Select](#)

8. We now need to associate a Training Definition. Click on **Add Training Definition**.

Define deep learning experiment details

Name
Single Convolution Layer on MNIST

Description
Experiment description

Machine Learning Service
Machine Learning Service
WatsonMachineLearning

Cloud Object Storage
bucket for storing training data files
Source: Connection to project COS / mnist-lab-train-bzb Update

Cloud Object Storage
bucket for storing results
Results: Connection to project COS / mnist-lab-results-bzb Update

Associate training definitions

Add training definition +

Name	Compute plan
No training definitions associated.	

☐ Use global execution command (override training definition values)

9. Click on **Existing training definition**, and select **Single Convolution Layer on MNIST**, select **1/2 x NVIDIA Tesla K80 (1 GPU)** for the compute plan, and then click **Select**.

New Existing

Select training definition

Existing training definitions
Single Convolution layer on MNIST

Attributes during this experiment

Compute configuration
1/2 x NVIDIA Tesla K80 (1 GPU)

Hyperparameter optimization method
none

Cancel Select

10. Click **Create and run**.

Define deep learning experiment details

Name

Single Convolution Layer on MNIST

Description

Experiment description

Machine Learning Service

Machine Learning Service

WatsonMachineLearning

Cloud Object Storage bucket for storing training data files

Source: Connection to project COS / mnist-lab-train-bzb Update

Cloud Object Storage bucket for storing results

Results: Connection to project COS / mnist-lab-results-bzb Update

Associate training definitions

Add training definition +

Name	Compute plan
Single Convolution layer on MNIST	1/2 x NVIDIA® Tesla® K80 (1 GPU)

☐ Use global execution command (override training definition values)

Cancel Create and run

Step 4: Monitor the Training Progress and Results

Training runs will be first queued, then in-process, and then completed. Use the **Training Runs** tab to keep track of progress. Usually, it will take between 3-5 minutes.

Queued Status

My Projects / Watson Studio Labs / Single Convolution Layer on MNIST

Single Convolution Layer on MNIST

Cancel runs in progress Add training runs

Training Runs Compare Runs Overview

1 Runs in total

0 hr, 0 min, 0 sec Total running time

Queued

NAME	SUBMITTED
Single Convolution layer on MNIST	0 hr, 0 min, 6 sec ago

In progress

NAME	DURATION
No training runs found.	

Completed

NAME	STATUS	DURATION	ACTIONS
No training runs found.			

In-Process Status

My Projects / Watson Studio Labs / Single Convolution Layer on MNIST

Single Convolution Layer on MNIST

Cancel runs in progress Add training runs

Training Runs Compare Runs Overview

1
Runs in total

0 hr, 0 min, 43 sec
Total running time

Queued

NAME	SUBMITTED
No training runs found.	

In progress

NAME	DURATION
Single Convolution layer on MNIST	0 hr, 0 min, 43 sec

Completed

NAME	STATUS	DURATION	ACTIONS
No training runs found.			

Completed Status – Note the statistics on the accuracy of the training set and validation set.

Single Convolution on MNIST

Add training runs

Training Runs Compare Runs Overview

1
Runs in total

0 hr, 2 min, 13 sec
Total running time

Queued

NAME	SUBMITTED
No training runs found.	

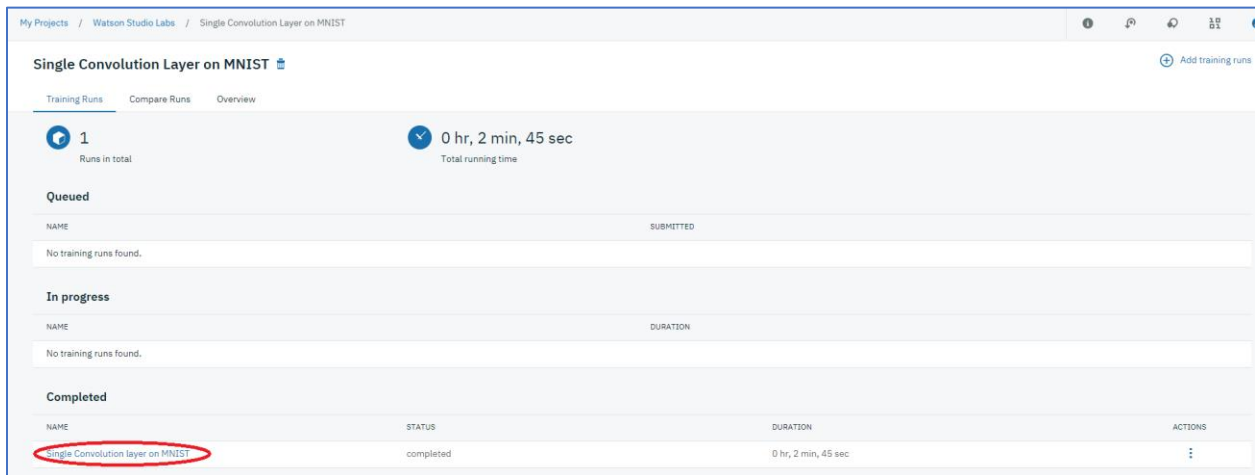
In progress

NAME	DURATION
No training runs found.	

Completed

NAME	STATUS	DURATION	ACC	LOSS	VAL_ACC	VAL_LOSS	ACTIONS
Single Convolution layer on MNIST	completed	0 hr, 2 min, 13 sec	0.994	0.019	0.976	0.108	⋮

1. When completed, click on the Single Convolution layer on MNIST link.



My Projects / Watson Studio Labs / Single Convolution Layer on MNIST

Single Convolution Layer on MNIST

Training Runs Compare Runs Overview

1 0 hr, 2 min, 45 sec
Runs in total Total running time

Queued

NAME	SUBMITTED
No training runs found.	

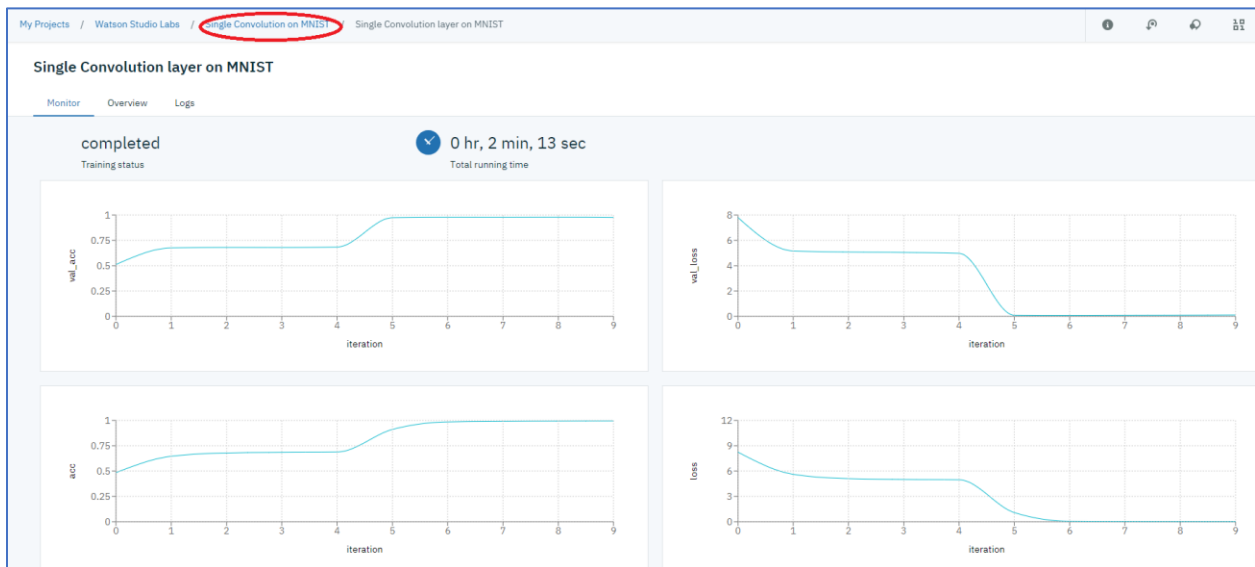
In progress

NAME	DURATION
No training runs found.	

Completed

NAME	STATUS	DURATION	ACTIONS
Single Convolution layer on MNIST	completed	0 hr, 2 min, 45 sec	

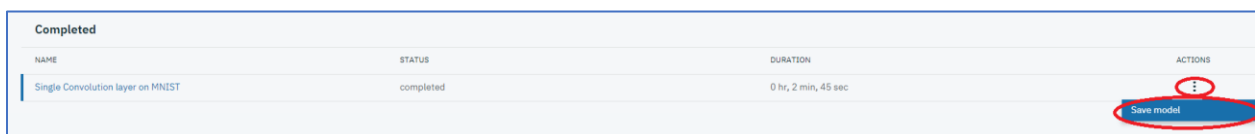
2. The display of the statistics over the training iterations is displayed. Click on the Single Convolution on MNIST tab to return to the Training Runs screen.



Step 5: Save and Deploy the Trained Model

We will now save the trained model to the Watson Machine Learning repository.

1. Click on the vertical ellipse under ACTIONS and click **Save model**.



Completed

NAME	STATUS	DURATION	ACTIONS
Single Convolution layer on MNIST	completed	0 hr, 2 min, 45 sec	

2. Enter a **Name** for the model (Single Convolution layer on MNIST) and click **Save**.

Save Model

Name
Single Convolutional layer on MNIST 65

Description
Model description 300

Cancel Save

3. Return to the Watson Studio **Assets** panel, by clicking on **Watson Studio Labs** in the breadcrumb path. Click on the **Assets** tab if the Assets panel is not showing.

IBM Watson Projects Tools Catalog Community Services

My Projects / Watson Studio Labs Single Convolution Layer on MNIST

✓ Model successfully saved. View model details [here](#).

4. Click on the newly saved model

Models New model

NAME	STATUS	TYPE	RUNTIME	LAST MODIFIED	ACTIONS
Single Convolutional Layer on MNIST	trained	tensorflow-1.5	python-3.5	5 Jun 2018	

5. Click on **Deployments**.

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes 'My Projects', 'Watson Studio Labs', and 'Single Convolutional layer on MNIST...'. The main header displays the project title 'Single Convolutional layer on MNIST' with a trash icon. Below the header, there are three tabs: 'Overview', 'Evaluation', and 'Deployments', with 'Deployments' being the active tab. A 'Summary' section follows, containing a table with the following details:

Machine learning service	Machine Learning
Model Type	tensorflow-1.5
Runtime environment	python-3.5
Training date	5 Jun 2018, 3:48 PM
Latest version	1c472928-e0ac-4146-9985-5b1c02bb8881

6. Click on **Add Deployment**.

This screenshot shows the 'Deployments' tab of the project. At the top, there are tabs for 'Overview', 'Evaluation', and 'Deployments'. Below the tabs, there is a table with columns for 'NAME', 'STATUS', 'DEPLOYMENT TYPE', and 'ACTIONS'. The table is currently empty, with a message 'Your model is not deployed.' displayed. In the top right corner of the table area, there is a button labeled 'Add Deployment' with a plus icon, which is circled in red.

7. Enter a **Name** (e.g. Single Convolution layer on MNIST Deployed), select **Web Service** (should be the default), and click on **Save**.

The screenshot shows the 'Create Deployment' form. It has a section titled 'Define deployment details' with the following fields:

- Name:** A text input field containing 'Single Convolutional layer on MNIST Deployed', which is circled in red.
- Description:** A text area for the deployment description.
- Deployment type:** A section with three radio button options: 'Web service' (selected and circled in red), 'Batch prediction', and 'Virtual (containers and files)'.

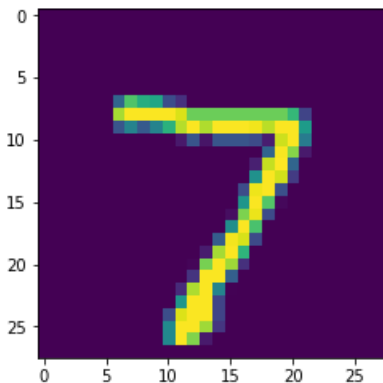
At the bottom right of the form, there are 'Cancel' and 'Save' buttons, with the 'Save' button circled in red.

8. The model is successfully deployed.

Overview Evaluation Deployments Lineage			
+ Add Deployment			
NAME	STATUS	DEPLOYMENT TYPE	ACTIONS
Single Convolution Layer on MNIST Deployed	DEPLOY_SUCCESS	Web Service	⋮

Step 6: Test the Deployed Model

We will now test the deployed model using the sample image data contained in the file test.json that was extracted from the mnist.zip file previously. The test.json file is a representation of the following image.



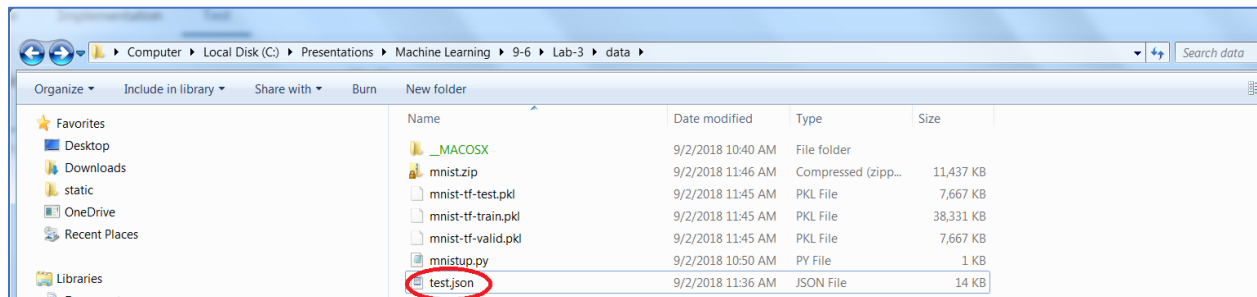
1. Click on the vertical ellipse, and then click on **View**.

Single Convolution on MNIST			
Overview Evaluation Deployments Lineage			
+ Add Deployment			
NAME	STATUS	DEPLOYMENT TYPE	ACTIONS
Single Convolution Layer on MNIST Deployed	DEPLOY_SUCCESS	Web Service	⋮
			View Delete

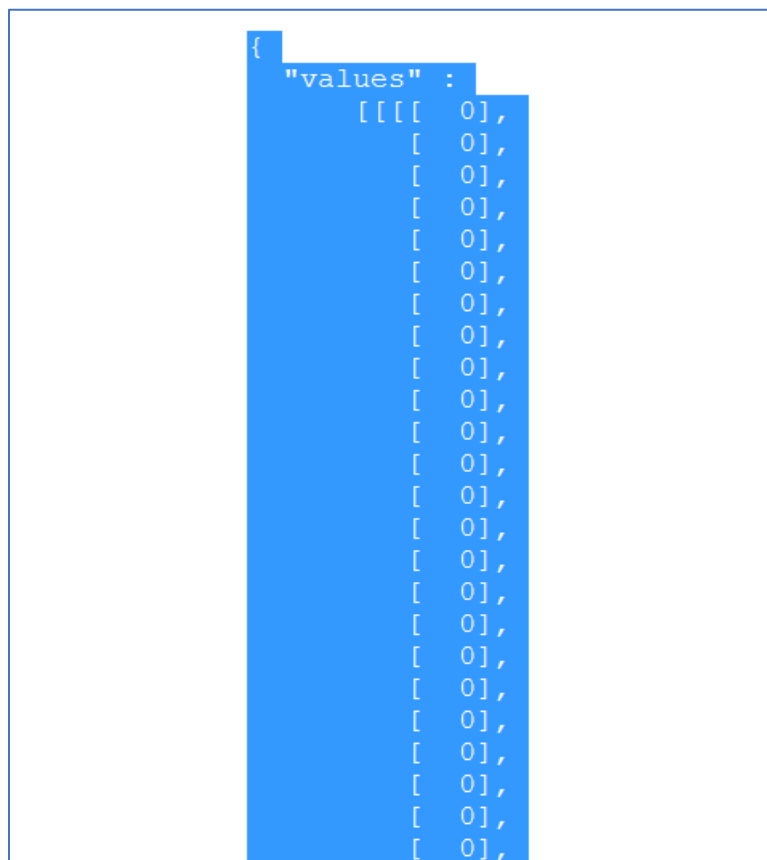
2. Click on **Test**.

Single Convolution Layer on MNIST Deployed	
Overview Implementation Test	
Deployment	
Name	Single Convolution Layer on MNIST Deployed
Type	Web Service
Deployment ID	89cf10e5-bd95-4b56-a728-e8b65fa98d83

- Go to the file directory where you have the “test.json” file stored, and double-click on the file.



4. Select the contents of the file by placing the cursor to the left of the { and pressing and holding the <Shift><Ctrl><End> keys.



- Copy and paste the content into the **Paste the request payload here** input data section. Make sure you have both the top bracket { at the beginning of the input data section and the bottom bracket } at the end of the input section data section, and then click on **Predict**.

Single Convolution Layer on MNIST Deployed

Overview Implementation **Test**

Enter input data

```
{  
  "values":  
    [[[[ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0],  
        [ 0]]]]
```

Predict

Single Convolution Layer on MNIST Deployed

Overview Implementation **Test**

Enter input data

```
[[[[ 0],  
    [ 0],  
    [ 0],  
    [ 0],  
    [ 0],  
    [ 0],  
    [ 0],  
    [ 0]]]]  
}
```

Predict

6. Each of the values represent the confidence level for the digits 0,1,2,3,4,5,6,7,8, and 9. The digit that is recognized would have the largest of the confidence levels. Based on the values returned, we can see that the number 7 would be selected as the best fit for this sample image which matches the image shown above.

Single Convolution Layer on MNIST Deployed

[Overview](#)[Implementation](#)[Test](#)

Enter input data

```
{
  "values":
    [[[[ 0],
        [ 0],
        [ 0],
        [ 0],
        [ 0],
        [ 0],
        [ 0]]]]
}
```

Predict

```
{
  "fields": [
    "prediction"
  ],
  "values": [
    [
      1.3815683352121771e-15,
      2.1411425399327925e-18,
      1.4536198037363307e-13,
      1.547869092014681e-13,
      3.484191859962678e-18,
      1.3316728982702908e-19,
      3.832952076167682e-23,
      1,
      1.7163898882906203e-13,
      4.053319832553193e-11
    ]
  ]
}
```

You have completed the Lab!

- ✓ Uploaded the data files to IBM Cloud Storage.
- ✓ Designed the neural network
- ✓ Trained the model
- ✓ Monitored the training progress and results
- ✓ Saved and Deployed the Trained Model
- ✓ Tested the Deployment