

Lab2: Amazon SageMaker Object Detection Algorithm

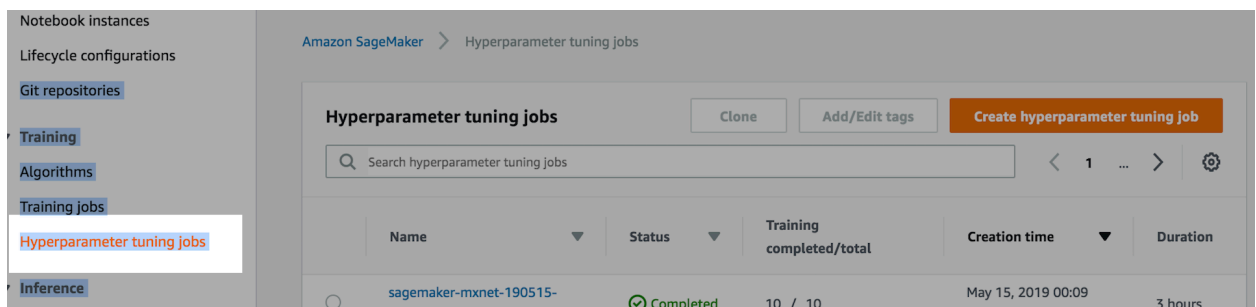
In this next lab we're going to train and deploy an object detection model using Amazon SageMaker's Object Detection algorithm.

This option has the benefit of allowing a data scientist to configure and tune an object detection model without having to write code. Amazon SageMaker provides a number of [built-in Algorithms](#). In some cases, they provide a performance and scalability benefit over other alternatives through GPU and distributed training support.

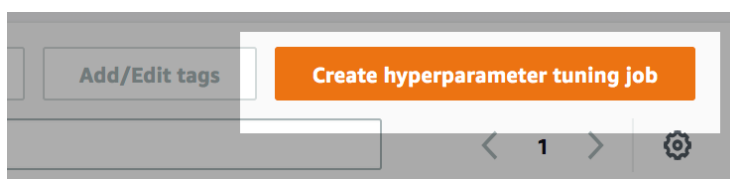
I. Launch an Automatic Model Tuning Job

Amazon SageMaker can automate much of the hyperparameter search process through its Automatic Model Tuning capabilities. In the following steps, we will provide Amazon SageMaker with a range of hyperparameters to search over. Amazon SageMaker will train multiple jobs on our behalf and in the process converge towards optimal hyperparameters within those ranges using [Bayesian Optimization](#). Bayesian Optimization has been demonstrated to be more effective than naïve approaches like grid and random search. Thus, the time and cost of training can be reduced through this capability.

1. From the Amazon SageMaker console, select **Hyperparameter tuning jobs** from the navigational panel on the left hand side.



2. Click on the “Create hyperparameter tuning job.”



3. Work through the job configuration wizard.

Enter a **unique name** for the job.

Job settings

Hyperparameter tuning job name

Amazon SageMaker adds this name to the name of training jobs launched by this tuning job. For example, for the name Failure-detection, the training job name is Failure-detection-xxxx-xxxx-xxxx-xxxx.

dtong-sv-workshop-ssd-hpo

The name must be from 1 to 32 characters and must be unique in your AWS account and AWS Region. Valid characters are a-z, A-Z, 0-9, and hyphen (-).

If you completed Lab 1, you can reuse the IAM role that you created previously under **“Use existing role.”** Otherwise, choose **“Create a new role”** and give the role access to all S3 buckets (for the sake of keep this lab exercise simple).

Amazon SageMaker adds this name to the name of training jobs launched by this tuning job. For

Create a new role
Enter a custom IAM role ARN
Use existing role
AmazonSageMaker-ExecutionRole-20171129T110981
AmazonSageMaker-ExecutionRole-20190508T145546
Choose an IAM role ▼

- Under Algorithm options, select **“Amazon SageMaker built-in algorithm.”**

Algorithm options

Use an Amazon SageMaker built-in algorithm, your own algorithm, or a third-party algorithm from AWS Marketplace.

▼ Algorithm source

- ☒ Amazon SageMaker built-in algorithm [Learn more](#)
- ☐ Your own algorithm resource
- ☐ Your own algorithm container in ECR [Learn more](#)
- ☐ An algorithm subscription from AWS Marketplace

Search and select **“Object Detection”** from the list of algorithms.

Image classification

IP Insights

Object Detection

Semantic Segmentation

Choose an algorithm or custom training image... ▼

5. Select **Pipe** under Input mode. This option allows the data to be streamed from S3 instead of staging the data on storage attached to the training machines. This mode reduces training start-up time, and is provides scalability necessary for large training sets.

▼ Choose an algorithm

Object Detection ▼

Container

The registry path where the training image is stored in Amazon ECR. [Learn more](#)

433757028032.dkr.ecr.us-west-2.amazonaws.com/object-detection:1

Input mode

You can provide your training data as a file or pipe.

Pipe ▼

Click on **Next**.

Cancel Next

6. You can optionally enable **early stopping**.

Early stopping

Early stopping stops training jobs when they are unlikely to improve the current best objective metric of the hyperparameter tuning job. [Learn more](#)

Training job early stopping type

Auto ▼

7. We need to specify an objective metric for Bayesian Optimization to optimize on. **Select validation:mAP and maximize**, so that the job seeks to maximizes mAP on our validation set.

Objective metric

To find the best training job, set an objective metric and tuning type. See the hyperparameter tuning job detail page for a summary of the best training job.

Strategy

The strategy that hyperparameter tuning uses to search over hyperparameter ranges that you specify.

Bayesian ▼

Objective metric

validation:mAP ▼

Type

maximize ▼

8. Next, configure the **parameters** for our Object Detection algorithm. Leave the parameters with their default values if they aren't explicitly mentioned below.

The parameters selected are practical for the purpose of demonstration. We're going to train the algorithm with only 10 images that were annotated in Lab 1.

In practice, we would use much more data, and require training time beyond the time available for a workshop.

Select **resnet-50** as our base network.

Name	Type	Scaling type	Value / Range
base_network	Static ▼	-	resnet-50 ▼

Change **num_classes** to 1 ("bird" is are only class).

num_classes	Static ▼	-	1 *
-------------	----------	---	-----

Set **epochs** to 60.

epochs	Static ▼	-	60
--------	----------	---	----

Set the **learning_rate** to the values below.

learning_rate	Continuous ▼	Logarithmic ▼	0.0001	-	0.0002
---------------	--------------	---------------	--------	---	--------

Reduce the number of optimizers down to **adam**.

optimizer	Categorical ▼	-	▼
adam ✕			

Set the **mini_batch_size** to be between 1 and 2 (since we only have 10 images).

mini_batch_size	Integer ▼	Linear ▼	1	-	2
-----------------	-----------	----------	---	---	---

Set **num_training_samples** to 10.

num_training_samples	Static ▼	-	10 *
----------------------	----------	---	------

Click on **Next** at the bottom of the page.

Cancel Previous Next

9. Configure the **train** channel, which specifies how the training data is fed into the training cluster.

- Set the input mode to **Pipe**.
- Set the **Record wrapper** to **RecordIO**.
- Set the **S3 data type** to **Augmented ManifestFile**. With this setting we can use the output.manifest file created in Lab 1 to inform the training cluster about how to locate, parse and stream the images on S3.
- Ensure you specify the attributes from the output.manifest file, so SageMaker knows how to parse the manifest:
 - The first attribute should be **source-ref** provides a mapping between image and annotations as well as the location in S3.
 - The second attribute should be the **name of your labeling job**. In the output.manifest file, the class labels and bounding box data are child elements. **This value will be different from the one in the screenshot provided below.**
- Set the S3 location to the S3 URI of your output.manifest.

▼ train

Channel name Input mode - optional

train Pipe ▼

Content type - optional

Compression type Record wrapper

None ▼ RecordIO ▼

S3 data type S3 data distribution type

AugmentedManifestFile ▼ FullyReplicated ▼

AugmentedManifestFile attribute names (Pipe input mode required)
Enter up to 16 AugmentedManifestFile JSON attribute names. [Learn more](#)

1. Remove ▲ ▼

2. Remove ▲ ▼

Add row

S3 location

10. Create a validation channel, and replicate the configurations from the train channel.

This will result in the training algorithm to use the training set for validation. In practice, we should have a separate data set for validation. For the sake of time and demonstration, we make do with the 10 images we annotated in Lab 1.

▼ validation

Channel name Input mode - optional

validation Pipe ▼

Content type - optional

Compression type Record wrapper

None RecordIO ▼

S3 data type S3 data distribution type

AugmentedManifestFile FullyReplicated ▼

AugmentedManifestFile attribute names (Pipe input mode required)
Enter up to 16 AugmentedManifestFile JSON attribute names. [Learn more](#)

1. source-ref Remove ⬆ ⬇

2. dtong-birds-labeling-job Remove ⬆ ⬇

Add row

S3 location

s3://dtong-cv-jumpstarter-workshop/birds/train/dtong-birds-labeling-job/manifests

11. Under **Output data configuration**, specify a location in your S3 bucket where the tuning process will output the trained model artifacts.

Output data configuration

S3 output path

s3://dtong-cv-jumpstarter-workshop/birds/train/hpo/out

To find a path, [go to Amazon S3](#)

Encryption key - optional
An encryption key protects your data. Type the key ID or key ARN that you want to use.

No Custom Encryption ▼

Click the **Next** button.

Cancel Previous Next

12. Under **Resource configurations**, select *ml.p3.2xlarge*. This provides us with 1 Nvidia V100 GPU. Leave the instance count to 1.

We utilize GPU for training because it will be magnitudes faster than CPU. We limit the instance count to and GPUs to 1 for the sake of managing the cost of this lab.

Resource configurations
The following resources will be applied to each training job.

Instance type

mlp3.2xlarge ▼

Instance count

1

Additional volume size per instance (GB)

1

Encryption key - *optional*

An encryption key protects your data. Type the key ID or key ARN that you want to use.

No Custom Encryption ▼

13. Set the **Maximum training jobs to 2**, and **parallel training jobs to 1**. In practice, you will run many more training jobs, so that Bayesian Optimization has an opportunity to converge towards optimal hyperparameters.

In the lab we only run two training jobs for the purpose of demonstration.

Resource limits
Specify resource limits for hyperparameter tuning and training jobs.

Maximum training jobs

The maximum number of training jobs that the hyperparameter tuning job can run.

2

A hyperparameter tuning job can run a maximum of 500 training jobs.

Maximum parallel training jobs

The maximum number of concurrent training jobs that the hyperparameter tuning job can run.

1

A hyperparameter tuning job can run a maximum of 10 parallel training jobs.

Maximum duration per training job

1

days ▼

A training job can run a maximum of 5 days.

Click on **Create jobs** to launch the tuning job.

Cancel

Previous

Create jobs

In practice, you will run multiple automatic tuning jobs. With each job, narrowing down the range of hyperparameters will yield better results. Use the warm-start feature to leverage the results from previous automatic model tuning jobs.

Create hyperparameter tuning job

Warm start - optional
Use the results from previous tuning jobs to improve the performance of this tuning job. [Learn more](#)

☒ **Enable warm start**

Warm start type

☒ **Identical data and algorithm**
Your input data and objective metric must be the same as your parent tuning jobs.

☐ **Transfer learning**
You may add additional data to your parent tuning jobs. The objective metric must remain the same.

Parent hyperparameter tuning job(s)
To improve this tuning job, use up to five previous hyperparameter tuning jobs and their training job results. Warm start is compatible with tuning jobs created after 10/01/18.

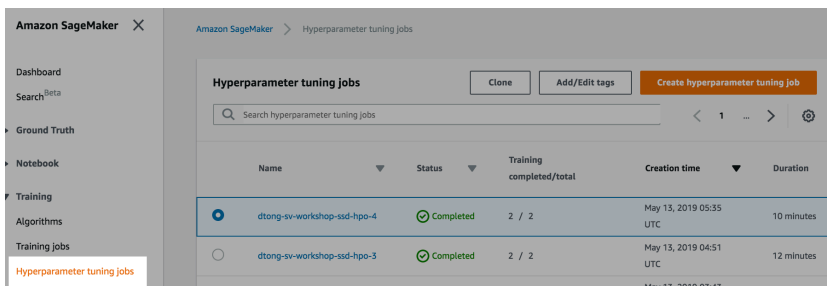
[Add hyperparameter tuning job](#)

Once you hone in on some good parameters, you may switch to manual control and fine tune the model by running individual jobs with appropriate parameters.

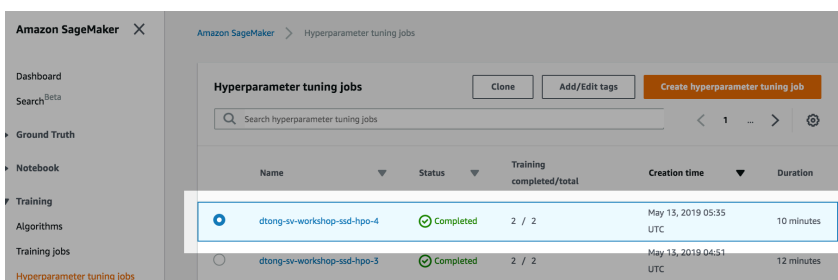
This tuning job will take 10-15 minutes to complete. This is a good time to take a break!

II. Deploy a Real-time Model Inference Endpoint

14. Select **Hyperparameter tuning jobs** on the navigation panel if you aren't already on the Hyperparameter tuning jobs page.



15. Select the hyperparameter tuning job that you created as shown in the screenshot below.



16. Scroll down. You should see something similar to the screenshot below. The tuning job should have two training jobs with corresponding objective metrics.

Best training job | **Training jobs** | Job configuration | Hyperparameter configuration | Tags

Training job status counter

Completed **2** | In Progress **0** | Stopped **0** | Failed **0** (Retryable: 0, Non-retryable: 0)

Training jobs
Sorting by objective metric value will display only jobs that have metric values.

View logs | View instance metrics | Stop | Create model

Search training jobs

Name	Status	Objective metric value	Creation time	Training Duration
dtong-sv-workshop-ssd-hpo-4-001-13f2ae27	Completed	0.7965668439865112	May 13, 2019 05:35 UTC	2 minute(s)
dtong-sv-workshop-ssd-hpo-4-002-858838c3	Completed	0.5731286406517029	May 13, 2019 05:39 UTC	3 minute(s)

Click on **Best training job** on the sub tab.

Best training job | Training jobs | Job configuration | Hyperparameter configuration | Tags

Best training job summary
This training job is the best training job for only this hyperparameter tuning job.

Create model

Name	Status	Objective metric	Value
dtong-sv-workshop-ssd-hpo-4-001-13f2ae27	Completed	validation:mAP	0.7965668439865112

Click on the **Create model** button. This will register the best model produced by the hyperparameter tuning job, and make it available for deployment via Amazon SageMaker.

Best training job | Training jobs | Job configuration | Hyperparameter configuration | Tags

Best training job summary
This training job is the best training job for only this hyperparameter tuning job.

Create model

Name	Status	Objective metric	Value
dtong-sv-workshop-ssd-hpo-4-001-13f2ae27	Completed	validation:mAP	0.7965668439865112

17. Work through the model configuration wizard.

Provide a **unique name** for your model.

Model settings

Model name

dtong-cv-workshop-sm-ssd

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

IAM role

Amazon SageMaker requires permissions to call other services on your behalf. Choose a role or let us create a role that has the [AmazonSageMakerFullAccess](#) IAM policy attached.

AmazonSageMaker-ExecutionRole-20190508T145546 ▼

Select the IAM role that you've been using to provide the model access to Amazon SageMaker and S3 resources.

Model settings

Model name

dtong-cv-workshop-sm-ssd

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

IAM role

Amazon SageMaker requires permissions to call other services on your behalf. Choose a role or let us create a role that has the [AmazonSageMakerFullAccess](#) IAM policy attached.

AmazonSageMaker-ExecutionRole-20190508T145546 ▼

Click on the **Create model** button. The model is now available for deployment.

Add container

▼ Tags - optional

Key	Value	
		Remove

Add tag

Cancel **Create model**

18. Switch to the **Endpoints** page by selecting **Endpoints** from the navigation pane:

Amazon SageMaker

Amazon SageMaker > Endpoints

Endpoints

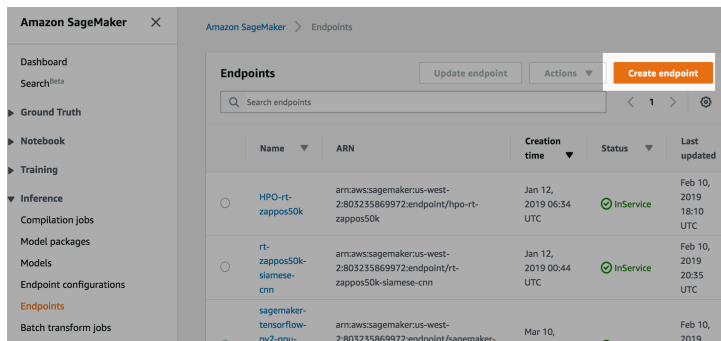
Update endpoint Actions **Create endpoint**

Search endpoints

	Name	ARN	Creation time	Status	Last updated
○	HPO-rt-zappos50k	arn:aws:sagemaker:us-west-2:80323569972:endpoint/hpo-rt-zappos50k	Jan 12, 2019 06:34 UTC	InService	Feb 10, 2019 18:10 UTC
○	rt-zappos50k-slamese-cnn	arn:aws:sagemaker:us-west-2:80323569972:endpoint/rt-zappos50k-slamese-cnn	Jan 12, 2019 00:44 UTC	InService	Feb 10, 2019 20:35 UTC
	sagemaker-tensorflow-	arn:aws:sagemaker:us-west-	Mar 10,		Feb 10,

Navigation pane: Dashboard, Search, Ground Truth, Notebook, Training, Inference, Compilation jobs, Model packages, Models, Endpoint configurations, **Endpoints**, Batch transform jobs

19. Click on the **Create endpoint** button.



20. Work through the Endpoint configuration form.

Provide a **unique name** for your endpoint.

Endpoint

Endpoint name
Your application uses this name to access this endpoint.

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Select **“Create a new endpoint configuration.”**

Attach endpoint configuration

☐ Use an existing endpoint configuration
Use an existing endpoint configuration or clone an endpoint configuration.

☒ Create a new endpoint configuration
Add models and configure the instance and initial weight for each model.

Provide a **unique name** for your endpoint configuration.

New endpoint configuration

To deploy models to Amazon SageMaker, first create an endpoint configuration. In the configuration, specify which models to deploy, and the relative traffic weighting and hardware requirements for each.

Endpoint configuration name

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Encryption key - optional
Encrypt your data. Choose an existing KMS key or enter a key's ARN.

Click on the Add model link.

Production variants

Model name	Training job	Variant name	Instance type	Elastic Inference	Initial instance count	Initial weight	Actions
There are currently no resources.							
Add model							
Create endpoint configuration							

Select the model that you registered previously, and click on the **Save** button.

Add model

< 1 ... >

Name	Creation time
<input checked="" type="radio"/> dtong-cv-workshop-sm-ssd	May 13, 2019 05:54 UTC
<input type="radio"/> sagemaker-mxnet-2019-05-11-19-01-30-771	May 11, 2019 19:06 UTC
<input type="radio"/> sagemaker-mxnet-2019-05-11-09-43-12-909	May 11, 2019 09:46 UTC
<input type="radio"/> batch-zappos50k-siamese-cnn	Jan 12, 2019 06:49 UTC
<input type="radio"/> sagemaker-pytorch-2019-01-12-06-34-16-780	Jan 12, 2019 06:34 UTC

[Cancel](#) [Save](#)

Click on the **Create endpoint configuration** button to finalize the creation of your endpoint configuration.

Production variants

Model name	Training job	Variant name	Instance type	Elastic Inference	Initial instance count	Initial weight	Actions
dtong-cv-workshop-sm-ssd		variant-name-1	ml.m4.xlarge	none	1	1	Edit
Add model							
Create endpoint configuration							

You should see confirmation that you endpoint configuration was successfully created.

✔ Success! You created an endpoint configuration. You can apply your endpoint configuration and create an endpoint.

New endpoint configuration

Change

Clone

Endpoint configuration name

dtong-cv-workshop-smssd-rt-ep-cfg

Encryption key

-

Production variants

Model name	Training job	Variant name	Instance type	Elastic Inference	Init inst cou
dtong-cv-workshop-sm-ssd	→	variant-name-1	ml.m4.xlarge	-	1

Click on **Create endpoint** to launch your inference endpoint.

Cancel

Create endpoint

You should see your endpoint in a “Creating...” status.

Endpoints

Update endpoint

Actions ▾

Create endpoint


Q



Search endpoints

<

1

>



	Name ▾	ARN	Creation time ▾	Status ▾	Last updated
	dtong-cv-workshop-smssd-rt-ep	arn:aws:sagemaker:us-west-2:803235869972:endpoint/dtong-cv-workshop-smssd-rt-ep	May 13, 2019 06:07 UTC	 Creating	May 13, 2019 06:07 UTC

It takes approximately 5 minutes for the endpoint to launch.

Endpoints

Update endpoint

Actions ▾

Create endpoint


Q



Search endpoints

<

1

>

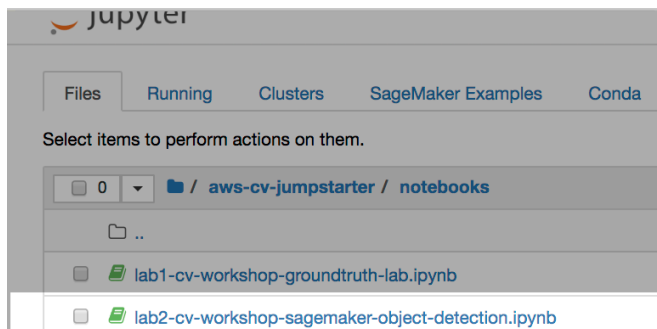


	Name ▾	ARN	Creation time ▾	Status ▾	Last updated
	dtong-cv-workshop-smssd-rt-ep	arn:aws:sagemaker:us-west-2:803235869972:endpoint/dtong-cv-workshop-smssd-rt-ep	May 13, 2019 06:07 UTC	 InService	May 13, 2019 06:15 UTC

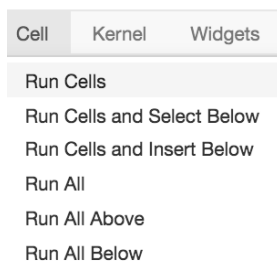
III. Test your Endpoint

In this section we're going to run some code in a Jupyter notebook to invoke our endpoint and visualize results.

21. **Log back on to** your SageMaker notebook instance. **Launch** the Jupyter notebook for **Lab 2** as shown in the screenshot below:



22. **Run** through each cell of the notebook.



There are a couple of places where you have to modify the script.

In the first cell, update the BUCKET variable with the name of the S3 bucket that you created in Lab 1.

```
BUCKET = '<<REPLACE WITH YOUR BUCKET NAME>>'
```

For instance,

```
BUCKET = 'dtong-cv-jumpstarter-workshop'
```

In the cell where you instantiate your real-time endpoint object, provide the name of the endpoint that you created in this lab.

```
RT_ENDPOINT_NAME = '<<REPLACE WITH THE NAME OF YOUR ENDPOINT>>'
```

For instance,

```
RT_ENDPOINT_NAME = 'dtong-cv-workshop-smssd-rt-ep'
```

23. **View the results.** The final cell will output images with predicted bounding boxes and classification.

Results will vary. Again, due to the purpose of demonstration, don't expect production grade results. In practice, we will need more data, training time and tuning.

```
In [54]: from os import walk  
  
img_dir = './images'  
for (dirpath, dirnames, filenames) in walk(img_dir):  
    for f in filenames:  
        show_bird_prediction(os.path.join(img_dir,f), od_endpoint.endpoint)
```

```
bird,0.6403394341468811  
Number of detections: 1
```

