**A Sample Project with AWS DeepLens**

This section provides a hands-on demonstration of a project created as part of an AWS DeepLens sponsored hack-a-thon. In this project, we use an AWS DeepLens device to do an *image classification*–based task. We train a model to detect if a piece of trash is from three potential classes: *landfill, compost,* or *recycling*.

**Summary**

AWS DeepLens is integrated with multiple AWS services. You use these services to create, train, and launch your AWS DeepLens project. To create any AWS DeepLens–based project you will need an AWS account.

Four key components are required for an AWS DeepLens–based project.

1. **Collect your data:** Collect data and store it in an Amazon S3 bucket.
2. **Train your model:** Use a Jupyter Notebook in Amazon SageMaker to train your model.
3. **Deploy your model:** Use AWS Lambda to deploy the trained model to your AWS DeepLens device.
4. **View model output:** Use Amazon IoT Greenrass to view your model's output after the model is deployed.

**Machine Learning workflow review**

The machine learning workflow contains several steps first introduced in Lesson 2. Let's briefly review the different steps and how they relate to the AWS DeepLens project.

1. **Define the problem.**
   * Using machine learning, we want to improve how trash is sorted. We're going to identify objects using a video stream, so we identify this as a computer vision–based problem.
   * We have access to data that already contains the labels, so we classify this as a *supervised learning* task.
2. **Build the dataset.**
   * Data is essential to any machine learning or computer vision–based project. Before going out and collecting lots of data, we investigate what kinds of data already exist and if they can be used for our application.
   * In this case, we have the data already collected and labeled.
3. **Train the model.**
   * Now that we have our data secured for this project, we use Amazon SageMaker to train our model. We cover specifics about this process in the demonstration video.
4. **Evaluate the model.**
   * Model training algorithms use **loss functions** to bring the model closer to its goals. The exact loss function and related details are outside the scope of this class, but the process is the same.
   * The loss function improves how well the model detects the different class images (compost, recycling, and landfill) while the model is being trained.
5. **Use the model.** 
   * We deploy our trained model to our AWS DeepLens device, where inference is performed locally.

**Demo: Using the AWS console to set up and deploy an AWS DeepLens project**

The following video demonstrates the end-to-end application (trash-sorting project) discussed in the previous video. This video shows you how to complete this project using the AWS console.

**Important**

* Storing data, training a model, and using AWS Lambda to deploy your model **incur costs** on your AWS account. For more information, see the [AWS account requirements](https://classroom.udacity.com/nanodegrees/nd065/parts/a5a4c41f-9cc7-48bd-9f00-582f35a7da53/modules/885b116b-2ca3-453a-8df1-4ea4b436b5da/lessons/8b79bd0c-6a77-40bc-8f96-b669c36d6103/concepts/2e2fcd70-fd26-45aa-9745-5fe947e8a3d9?contentVersion=1.0.0&contentLocale=en-us) page.
* You are **not required** to follow this demo on the AWS console. However, we recommend you watch it and understand the flow of completing a computer vision project with AWS DeepLens.

**Demo Part 1: Getting Setup and Running the Code**

Click [here](https://video.udacity-data.com/topher/2021/May/609ab503_aws-deeplens-custom-trash-detector/aws-deeplens-custom-trash-detector.ipynb) to download the Jupyer notebook the instructor used in the demo.

**Summary: demo part 1**

In this demo, you first saw how you can use Amazon S3 to store the image data needed for training your computer vision model. Then, you saw how to use Amazon SageMaker to train your model using a Jupyter Notebook

**Demo Part 2: Deployment and Testing**

**Summary: demo part 2**

Next, you used AWS Lambda to deploy your model onto an AWS DeepLens device. Finally, once your model has been deployed to your device, you can use AWS IoT Greengrass to view the inference output from your model actively running on your AWS DeepLens device.

**More Projects on AWS DeepLens and Other AWS Services**

* In [this blog post on the AWS Machine Learning blog](https://aws.amazon.com/blogs/machine-learning/protecting-people-through-virtual-boundaries-computer-vision/), you learn about how computer vision–based applications can be used to protect workers in workplaces with autonomous robots. The post demonstrates you how you can create a virtual boundary using a computer and AWS DeepLens.
* Using [Amazon Rekognition](https://aws.amazon.com/rekognition/?blog-cards.sort-by=item.additionalFields.createdDate&blog-cards.sort-order=desc&utm_source=Udacity&utm_medium=Webpage&utm_campaign=Udacity%20AWS%20ML%20Foundations%20Course) and AWS DeepLens, you can [create an application](https://aws.amazon.com/blogs/machine-learning/building-a-smart-garage-door-opener-with-aws-deeplens-and-amazon-rekognition/) that uses OCR or optical character recognition to recognize a car's license plate, and open a garage door.
* You can use [Amazon Alexa and AWS DeepLens to create a Pictionary style game](https://www.awsdeeplens.recipes/300_intermediate/330_guess_drawing/?utm_source=Udacity&utm_medium=Webpage&utm_campaign=Udacity%20AWS%20ML%20Foundations%20Course). First, you deploy a trained model to your AWS DeepLens which can recognize sketches drawn on a whiteboard and pair it with an Alexa skill that serves as the official scorekeeper.

**Supporting Materials**

* [Aws-Deeplens-Custom-Trash-Detector](https://video.udacity-data.com/topher/2021/May/609ab503_aws-deeplens-custom-trash-detector/aws-deeplens-custom-trash-detector.ipynb)

;