Distributed Training of Deep Networks on Amazon Web Services* (AWS)

Published on August 12, 2016 Translate

[Download Document (/sites/default/files/managed/61/28/DistributedTrainingofDeepNetworksonAmazonWebServices.pdf)]

Ravi Panchumarthy (Intel), Thomas “Elvis” Jones (AWS), Andres Rodriguez (Intel), Joseph Spisak (Intel)

Deep neural networks are capable of amazing levels of representation power resulting in state-of-the-art accuracy in areas such as computer vision, speech recognition, natural language processing, and various data analytic domains. Deep networks require large amounts of computation to train, and the time to train is often days or weeks. Intel is optimizing popular frameworks such as Caffe*, TensorFlow*, Theano*, and others to significantly improve performance and reduce the overall time to train on a single node. In addition, Intel is adding or enhancing multi. node distributed training capabilities to these frameworks to share the computational requirements across multiple nodes and further reduce time to train. A workload that previously required days can now be trained in a matter of hours. Read more about this (/en-us/articles/caffe-scoring-optimization-intel-xeon-processor-e5-series).

Amazon Web Services* (AWS) Virtual Private Cloud (VPC) provides a great environment to facilitate multinode distributed deep network training. AWS and Intel partnered to create a simple set of scripts for creating clusters that allows developers to easily deploy and train deep networks, leveraging the scale of AWS. In this article, we provide the steps to set up the AWS CloudFormation* environment to train deep networks using the Caffe network.

AWS CloudFormation Setup

The following steps create a VPC that has an Elastic Compute Cloud (EC2) t2.micro instance as the AWS CloudFormation cluster (cfncluster) controller. The cfncluster controller is then used to create a cluster composed of a master EC2 instance and a number of compute EC2 instances within the VPC.

Steps to deploy the Cloudformation and cfncluster

1. Use the AWS Management Console to launch the AWS CloudFormation (Figure 1).

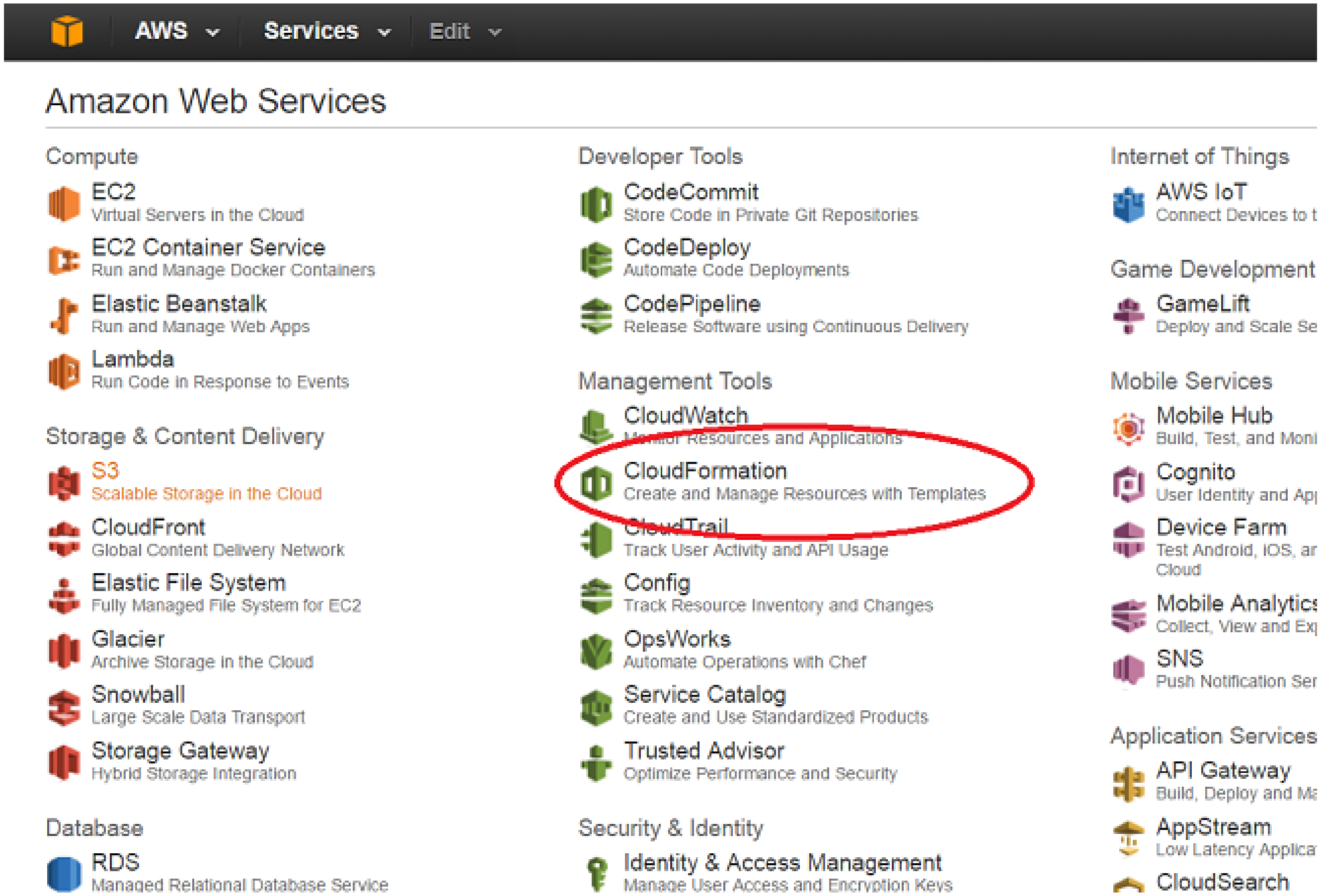

Figure 1. CloudFormation in Amazon Web Services

2. Click **Create Stack**.
3. In the section labeled, Choose a template (Figure 2), select **Specify an Amazon S3 template URL**, and then enter [https://s3.amazonaws.com/caffecfncluster/1.0/intelcaffe_cfncluster.template (https://s3.amazonaws.com/caffecfncluster/1.0/intelcaffe_cfncluster.template)](https://s3.amazonaws.com/caffecfncluster/1.0/intelcaffe_cfncluster.template). Click **Next.**

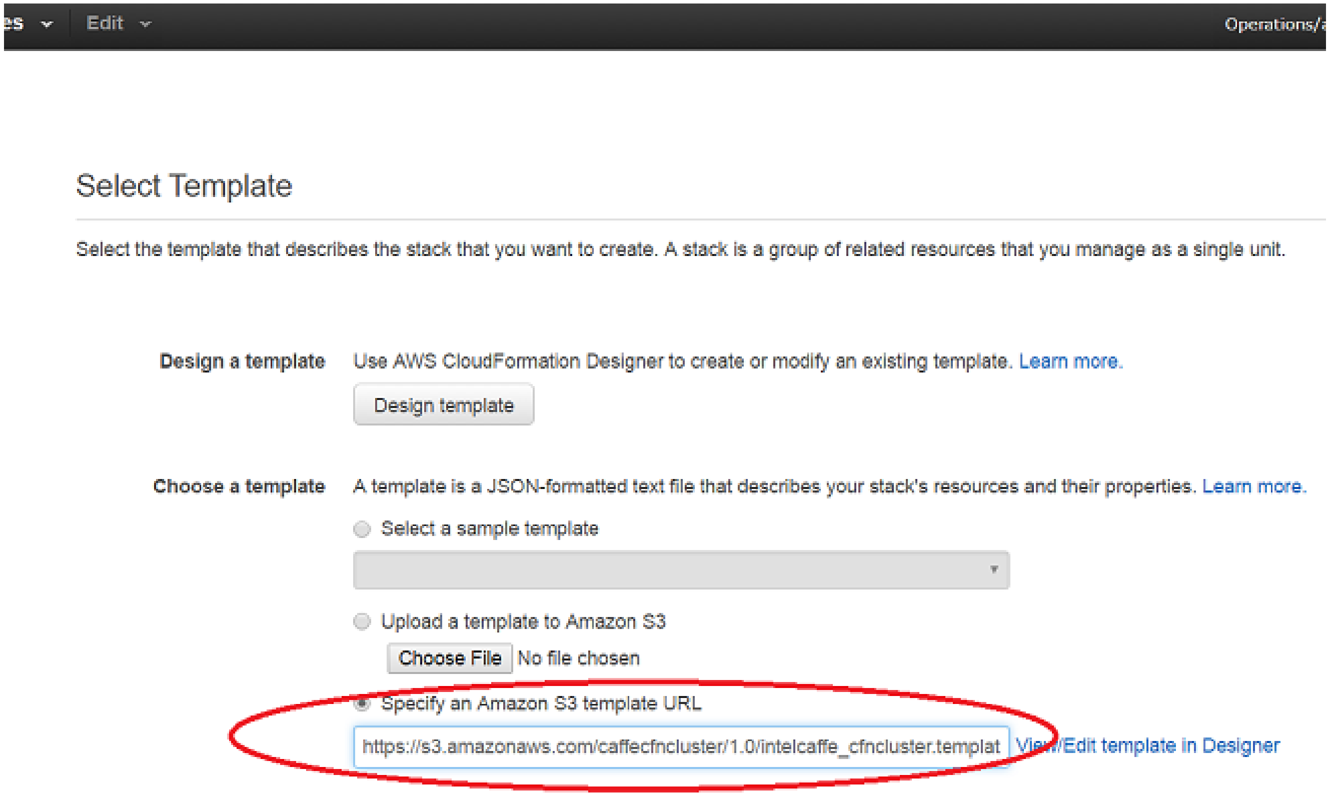

Figure 2. Entering the template URL.

4. Give the Stack a name, such as myFirstStack. UnderSelect a key pair, find the key pair you just named ([follow these instructions (http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair)](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair) if you need to create a key pair). Leave the rest of the Parameters as they are. Click **Next**.
5. Enter a Key, for example, `name`, and a Value, such as, `cfnclustercaffe`.

   Note that you can give any names to the key and value. The name does not have to match the key-pair from the previous step.
6. Click **Next**.
7. Review the stack, check the acknowledgement box, and then click **Create**. Creating the stacks will take a few minutes. Wait until the status of all three created stacks is CREATE_COMPLETE.
8. The template used in Step 3 calls two other nested templates, creating a VPC with an EC2 t2.micro instance (Figure 3). Select the stack with the EC2 instance, and then select **Resources**. Click the Physical ID of the cfnclusterMaster.

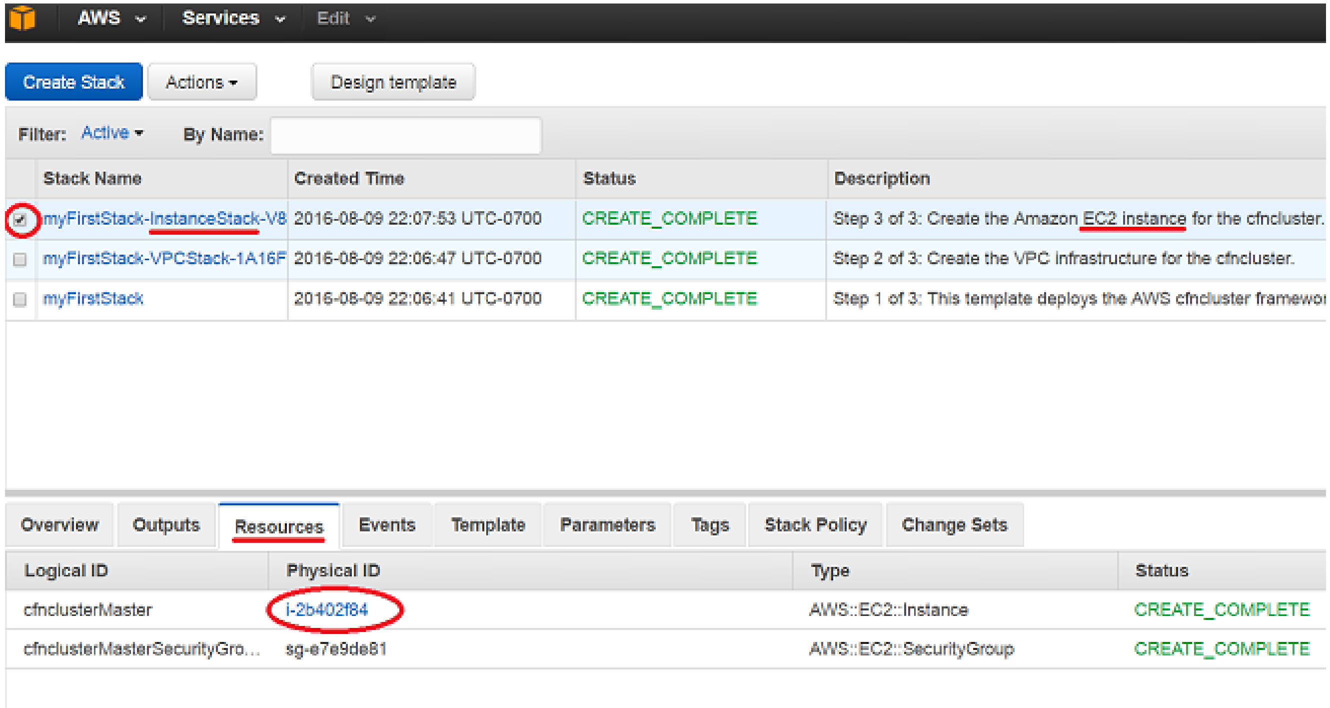

Figure 3. Selecting the Physical ID from the Resources tab.

9. This will take you to AWS EC2 console (Figure 4). Under Description, note the VPC ID and the Subnet ID as you'll need them in a later step. Right-click on the instance, select **Connect** and follow the instructions.

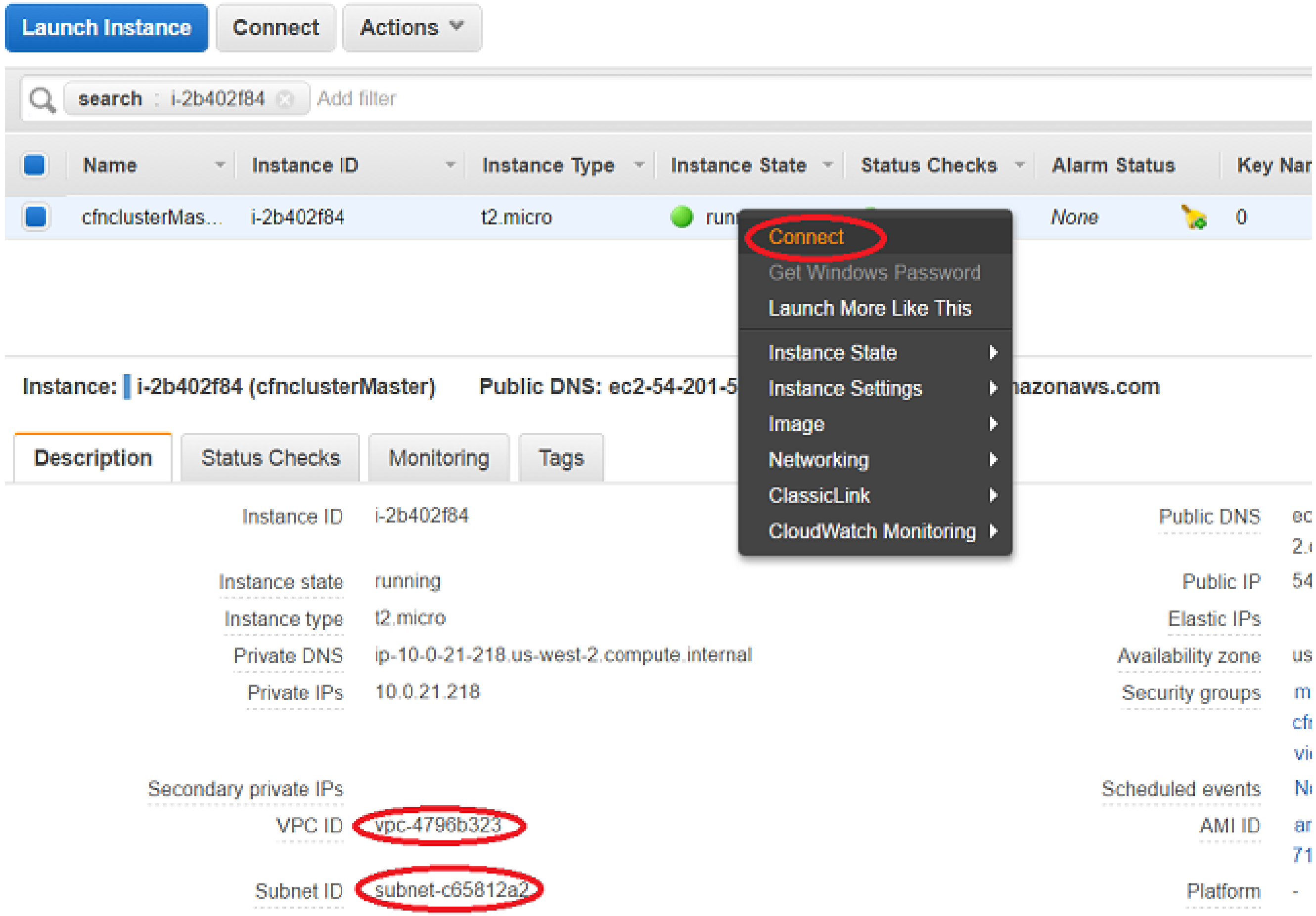

Figure 4. AWS EC2 console.

10. Once you ssh into the instance, prepare to modify the cluster's configuration with the following commands:

```
cd .cfncluster

cp config.edit_this_cfncluster_config config

vi config
```

11. Follow the comments in the config file (opened with the final command in Step 9) to fill in the appropriate information.

- Obtain your AWS_ACCESS_KEY and AWS_SECRET_ACCESS_KEY from your AWS system administrator if you don't have one already or [follow these steps to obtain them (http://blogs.aws.amazon.com/security/post/Tx1R9KDN9ISZ0HF/Where-s-my-secret-access-key)](http://blogs.aws.amazon.com/security/post/Tx1R9KDN9ISZ0HF/Where-s-my-secret-access-key).

Note that while the master node is not labelled as a compute node, it also acts as a compute node. Therefore, if the total number of nodes to be used in training is 32, then choose a queue_size = 31 compute nodes.

- Use the VPC ID and Subnet ID obtained in Step 8.
- The latest custom_ami to use should be `ami-77aa6117`; this article will be updated when newer AMI are provided.

12. Launch a cluster with the command: `cfncluster create <vpc_name_choosen_in_config_file>`. This will launch more AWS CloudFormation templates. You can see them via the AWS CloudFormation page in the AWS Management Console.

Sample Scripts to Train a Few Popular Networks

After the cloud-formation-setup is complete, if you configured the size of the cluster to be *N*, there will be *N*+1 instances created (1 master node and *N* compute nodes). Note that the master node is also treated as a compute node. The created cluster has a shared drive among all *N*+1 instances. The instances contain intelcaffe, Intel® Math Kernel Library (Intel® MKL) and sample scripts to train CIFAR-10 and GoogLeNet. To start training a sample network, login into the master node.

To start training a CIFAR-10 model with provided solver and train_val prototxt files, run:

```
cd ~/scripts/
./aws_ic_mn_run_cifar.sh
```

To start training a GoogLeNet model, you should download ImageNet dataset and configure the variables `path_to_imagenet_train_folder`, `batchsize_pernode` and others if required in the script and run the `./aws_ic_mn_run_googlenet.sh script`:

```
cd ~/scripts/
#Edit variables path_to_imagenet_train_folder, batchsize_pernode and others
if required
vi ./aws_ic_mn_run_googlenet.sh
./aws_ic_mn_run_googlenet.sh
```

The script `aws_ic_mn_run_cifar.sh` creates a hosts file (`~/hosts.aws`) which contains all the IP addresses of the instances in your VPC. It then updates the solver and train_val prototxt files located in `~/models/cifar10/`. You could modify these prototxt files to suit your training requirements. The `aws_ic_mn_run_cifar.sh` script will start the data server, which will provide data to the compute nodes. There will be a little overhead on the master with data server running along with the compute. After the data server is launched, the distributed training is launched using the mpirun command.

The script `aws_ic_mn_run_googlenet.sh` creates a hosts file (`~/hosts.aws`) which contains all the IP addresses of the instances in your VPC. Unlike, the CIFAR-10 example where the data server provides the data, in GoogLeNet training, each worker will read its own data. The script will create separate solver, train_val prototxt files and train.txt files for each worker based on the template solver and train_val prototxt located in `~/models/googlenet/`. You could modify these template prototxt files to suit your training requirements. The `aws_ic_mn_run_googlenet.sh` script will then launch the job using the mpirun command.

Notices

1 comment

Top

Ginsberg, Ziv (/en-us/user/1489257) said on Oct 18,2016

(/en-us/user/1489257)

Defently gonna find time for play with it

Add a Comment

Sign in (/en-us/user/login?language=en-us&destination=node/675082&https=1)

Have a technical question? Visit our forums (/en-us/forum).

Have site or software product issues? Contact support (/en-us/contact).