1. SageMaker Instance

I chose ml.t3.medium for my notebook instance type because its hourly rate is only $0.0416 and it is sufficient to run my notebook. I used the ml.m5.xlarge for training since it is suitable for training an image classification model.

single instance：pytorch-inference-2022-04-27-05-44-50-366

1. EC2 Instance

I chose t2.xlarge instance and the Deep Learning AMI (Amazon Linux 2) Version 60.0. This is a reasonable balance of performance and affordability. T2 instances can satisfy high CPU performance and I don’t need a lot of computing and t2.xlarge is enough.

1. Difference between EC2 script and SageMaker script
2. The code in the EC2 script is responsible for saving the model to the local path, while the SageMaker script handled internally and the model data was stored to a S3 location.
3. In the EC2, all hyperparameters and output locations are mentioned directly.
4. Ec2 script does not have main function.
5. Lambda functions

Since we created two endpoints in Step 1, one for single instance training and the other for multi-instance training. For this lambda function, I used multi instance trained endpoint.

The lambda function will accept image inputs in json format and invoke an endpoint that returns a response with a prediction. The function then packages information such as status code and body into a dictionary and returns it as the response of the function.

Prediction scores from Lambda Function

[0.3312390446662903, 0.1241925209723496, -0.6577092409133911, -1.0731314420700073, -2.2706263065338135, -3.5305466651916504, 0.10673454403877258, -1.0987365245819092, -2.7904741764068604, 2.3420016765594482, 0.8868023157119751, -2.4440958499908447, -1.085627555847168, 0.9981246590614319, -2.4145634174346924, -2.5615897178649902, -2.6598927974700928, -1.2003998756408691, -3.1477959156036377, 0.5177315473556519, -1.4471185207366943, -0.908940851688385, -3.3537514209747314, -3.410036563873291, -2.358534574508667, -4.360621929168701, -2.004969596862793, -2.377509593963623, -1.269318699836731, -1.0121127367019653, -0.5772803425788879, -2.1388115882873535, -4.716067790985107, -0.023615673184394836, -2.239001989364624, -3.6613147258758545, -2.8146815299987793, -1.864284634590149, 1.1093313694000244, -1.3443825244903564, -1.6215791702270508, -2.132383346557617, 1.0124493837356567, -2.4523706436157227, -0.16058950126171112, -2.5105860233306885, -2.4992122650146484, 0.19574496150016785, -1.0972182750701904, -2.5561790466308594, -2.105484962463379, -3.2779054641723633, -4.7049946784973145, -1.3346025943756104, -3.0439774990081787, -1.0216214656829834, -2.3901243209838867, -4.023900508880615, -1.594041109085083, -0.8026221394538879, -3.734494209289551, -0.9273689389228821, -1.8518251180648804, -3.1978719234466553,

-2.069045305252075, -2.1380717754364014, 0.6809729933738708, -2.3260533809661865, -0.39181116223335266, 0.1527157574892044, -0.5785276889801025, -1.7788881063461304, -3.9772324562072754, -2.0746607780456543, -3.0547993183135986, -0.7032391428947449, -4.057677268981934, -0.6666017174720764, -4.368079662322998, -4.516876220703125, 2.878140687942505, -4.181717395782471, -0.9569286108016968, -0.43563368916511536, -1.9721391201019287, -1.2047059535980225, -0.8335296511650085, -3.868185520172119, -3.004767417907715, -0.03276382386684418, -4.781686782836914, -3.6673121452331543, -4.848998546600342, -4.433956146240234, -2.640279531478882, -0.8203840255737305, -1.063960313796997, -1.5861718654632568, -3.8939929008483887, -3.215728521347046, -3.1812422275543213, -0.12406864762306213, -0.9839164614677429, -2.4763545989990234, -2.6550557613372803, -3.057565212249756, -3.243168354034424, -1.2994898557662964, -1.1153687238693237, 0.20982450246810913, -1.7602503299713135, -0.536587655544281, -3.2850821018218994, -2.2085304260253906, -1.1557217836380005, 0.3915814757347107, -2.0830204486846924, -1.6466426849365234, -4.630002021789551, -1.0387139320373535, -2.360456943511963, -2.546125888824463, -2.5815792083740234, -0.8240213990211487, -5.775858402252197, -2.7869718074798584, -2.7823057174682617, 0.06473562866449356, -2.560988187789917, -2.4633893966674805, -4.129008769989014, -0.25464266538619995, -2.1182408332824707]

1. Security

When creating a new role for the lambda functions, I got accessdenied exception due to the fact that lambda function did not have access to SageMaker endpoint. So we have to add “SageMakerFullAccess” policy role to the lambda functions, this could cause vulnerabilities.

1. Concurrency and Autoscaling

For concurrency, I chose provisioned concurrency because in general, this image classification project may receive a lot of traffic that we could not predict at this time and the ability to handle concurrency seems efficient.

For autoscaling, I set the Maximum Instance Count to 3 since I used m5.large instance type to train the model. For the built-in scaling policy, I set the Target Value to 10 and both scale-in and scale-out cool down to be 30 seconds.