SYNOPSIS

**ABSTRACT**Spot price or variable price resources are the recent advancement in cloud computing business models. The spot pricing mechanism follows auction-based cloud model in which the price of spot instances changes with time. If the termination of instance is initiated by user before completion of an hour, then user is bound to pay for the complete hour. However, if the instance termination is initiated by Amazon then the user doesn’t have to pay for the partial hour. The termination of instance is done by cloud provider without any prior notification to the user when the bid price of user gets down to the current spot price. This poses a severe limitation to the applications where the time of availability is very important. Thus, it becomes very important for the users to predict spot prices before placing their bids. Amazon web services provides different pricing models pay-per-use, fixed, and auction-based (spot price). It is seen that the spot price is a minimum of 5 times cheaper than the other pricing models but there is no guarantee that you will be given the instance. It depends on the price you bid. Thus, analysis of historical data for spot price in order to efficiently (minimal cost) schedule the jobs is important.

Spot-instance pricing makes high-performance GPUs much more affordable for deep learning researchers and developers who run training jobs that span several hours or days. Spot instances allow you to access spare Amazon EC2 compute capacity at a steep discount compared to on-demand rates. Spot instances are great for deep learning workflows.

**INTRODUCTION**The distinguishing feature of spot instance is its dynamic pricing. The prices of spot instances vary dynamically with time based on demand and supply of cloud resources in the datacenters across the globe. Customers place bids to obtain spot instances using an online auction platform. The auction platform determines the market clearance price also known as spot price and if the users bid above the aforementioned price, they obtain the instances. Cloud vendors provide current and archived spot price data so as to assist their customers in bidding process. Amazon Web Services (AWS) provide a web-based API access to the users for bidding spot instances. The spot instance bid request consists of the following parameters: 1) Instance type 2) Number of instances 3) Availability zone 4) Bid amount of user i.e. price/instance/per hour.

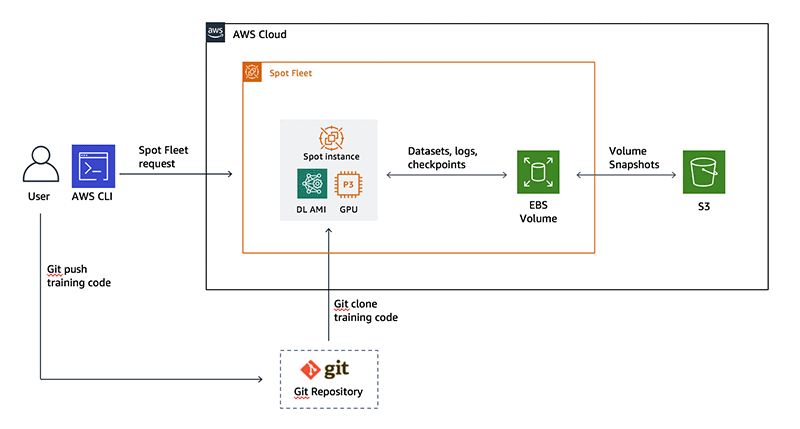
**OBJECTIVES**

Variable pricing cloud resources are the most recent advancement in cloud computing business models. Cloud vendors like Amazon Web Services, Amazon AWS provide a new cloud instance type known as 'Spot instance'. The distinguishing feature of spot instance is its dynamic pricing. The price of spot instances varies dynamically with time based on demand and supply of cloud resources in the data centers across the globe. Customers place bids to obtain spot instances using an online auction platform. The auction platform determines the market clearance price, 'Spot price' and the users whose bids are above the aforementioned price obtain the instances. Cloud vendors provide current and archived spot price data to assist their customers in bidding process. The major challenge for the customers in this new business model is to predict the spot price before placing their bids. We are to take the historical data and predict the instances. Moreover, you can easily combine Spot Instances with On-Demand and RIs to further optimize workload cost with performance. Due to the operating scale of AWS, Spot Instances can offer the scale and cost savings to run hyper-scale workloads. You also have the option to hibernate, stop or terminate your Spot Instances when EC2 reclaims the capacity back with two-minutes of notice. Only on AWS, you have easy access to unused compute capacity at such massive scale – all at up to a 90% discount. Spot instances are great for deep learning workflows, but there are a few challenges associated using spot instances versus on-demand instances. First, spot instances can be preempted and can be terminated with just 2 minutes notice. This means you can’t count on your instance to run a training job to completion. Therefore, it’s not recommended for time-sensitive workloads. Second, instance termination can cause data loss if the training progress is not saved properly. Third, if you decide your application should not be interrupted after launching the spot instance, your only option is to stop the spot instance and re-launch as an on-demand or reserved instance.

**METHODOLOGY**

To address the challenges, here is a how to set up spot instances for deep learning training workflows while minimizing training progress loss if a spot interruption occurs. Our goal is to implement a setup with the following characteristics:

* Decouple compute, storage and code artifacts, and keep the compute instance stateless. This enables easy recovery and training state restore when an instance is terminated and replaced
* Use a dedicated volume for datasets, training progress (checkpoints) and logs. This volume should be persistent and not be affected by instance termination
* Use a version control system (e.g. Git) for training code. This repo should be cloned to commence/resume training. this enables traceability and prevents loss of code changes when instance is terminated
* Minimize code changes to the training script. This ensures that the training script can be developed independently, and backup and snapshot operations are performed outside of the training code
* Automate, automate, automate. Automate replacement instance creation after termination, attaching of dataset and checkpoints EBS volume at launch, moving volumes across Availability Zones, performing instance state restore, resuming training, and terminating instance once training is finished.



**HARDWARE/SOFTWARE REQUIREMENTS**

**Software**: Python 3, Jupyter Notebook, Machine Learning libraries (Numpy,Pandas, Scilearn), AWS Account, AWS CLI tool, Tensorflow

**Hardware**: 8 GB minimum, linux/windows, intel i5 or above.