EXAM PROFESSIONAL MACHINE LEARNING ENGINEER TOPIC 1 QUESTION 151 DISCUSSI...

Actual exam question from Google's Professional Machine Learning Engineer
Question #: 151
Topic #: 1
[All Professional Machine Learning Engineer Questions]

While running a model training pipeline on Vertex AI, you discover that the evaluation step is failing because of an out-of-memory error. You are currently using TensorFlow Model Analysis (TFMA) with a standard Evaluator TensorFlow Extended (TFX) pipeline component for the evaluation step. You want to stabilize the pipeline without downgrading the evaluation quality while minimizing infrastructure overhead. What should you do?

A. Include the flag -runner=DataflowRunner in beam_pipeline_args to run the evaluation step on Dataflow.

B. Move the evaluation step out of your pipeline and run it on custom Compute Engine VMs with sufficient memory.

C. Migrate your pipeline to Kubeflow hosted on Google Kubernetes Engine, and specify the appropriate node parameters for the evaluation step.

D. Add tfma.MetricsSpec () to limit the number of metrics in the evaluation step.

Show Suggested Answer

by RaghavAI at *Feb. 5, 2023, 4:15 p.m.*

Comments

MultipleWorkerMirroredStrategy Highly Voted 1 year ago

Selected Answer: A

"Evaluator leverages the TensorFlow Model Analysis library to perform the analysis, which in turn use Apache Beam for scalable processing." Since Dataflow is Google Cloud's serverless Apache Beam offering, this option can easily be implemented to address the issue while leaving the evaluation logic as such identical

https://www.tensorflow.org/tfx/guide/evaluator#evaluator_and_tensorflow_model_analysis

upvoted 6 times

pico 11 months, 3 weeks ago

If we have to add dataflow then this condition is not met: minimizing infrastructure overhead

upvoted 1 times

Zepopo 7 months, 2 weeks ago

No, it is. If we choose another option, there would be:
B - you need to configure VMs and migrate all workloads
C - also overhead with migrating
D - downgrading the evaluation quality
So just switch runner seems a very easy option

upvoted 2 times

M25 Highly Voted 1 year, 5 months ago

Selected Answer: A

Links already provided below:
“That works fine for one hundred records, but what if the goal was to process all 187,002,0025 rows in the dataset? For this, the pipeline is switched from the DirectRunner to the production Dataflow runner.” [Option A]
https://blog.tensorflow.org/2020/03/tensorflow-extended-tfx-using-apache-beam-large-scale-data-processing.html.
"Metrics to configure (only required if additional metrics are being added outside of those saved with the model).”
https://www.tensorflow.org/tfx/guide/evaluator#using_the_evaluator_component
will thus add, not “limit the number of metrics in the evaluation step”. [Option D]

upvoted 5 times

gscharly Most Recent 6 months, 2 weeks ago

Selected Answer: A

with D we're downgrading evaluation. Dataflow is serverless so no infrastructure overhead is included

upvoted 2 times

pico 11 months, 3 weeks ago

Selected Answer: D

Limiting Metrics: TensorFlow Model Analysis (TFMA) allows you to define a subset of metrics that you are interested in during the evaluation step. By using tfma.MetricsSpec(), you can specify a subset of metrics to be computed during the evaluation, which can help reduce the memory requirements.

Out-of-Memory Error: Out-of-memory errors during model evaluation often occur when the system is trying to compute and store a large number of metrics, especially if the model or dataset is large. By limiting the number of metrics using tfma.MetricsSpec(), you can potentially reduce the memory footprint and resolve the out-of-memory error.

upvoted 2 times

PST21 1 year, 3 months ago

Based on the question's context, the correct option to stabilize the pipeline without downgrading the evaluation quality while minimizing infrastructure overhead is:

D. Add tfma.MetricsSpec() to limit the number of metrics in the evaluation step.

The question specifies that the evaluation step is failing due to an out-of-memory error. In such a scenario, limiting the number of metrics to be computed during evaluation using tfma.MetricsSpec() can help reduce memory requirements and potentially resolve the out-of-memory issue.

upvoted 1 times

tavva_prudhvi 1 year, 4 months ago

Selected Answer: D

By adding tfma.MetricsSpec(), you can limit the number of metrics that are computed during the evaluation step, thus reducing the memory requirement. This will help stabilize the pipeline without downgrading the evaluation quality, while minimizing infrastructure overhead. This option is a quick and easy solution that can be implemented without significant changes to the pipeline or infrastructure.

Option A: Including the flag -runner=DataflowRunner in beam_pipeline_args to run the evaluation step on Dataflow may help to increase memory availability, but it may also increase infrastructure overhead.

upvoted 1 times

tavva_prudhvi 1 year, 3 months ago

it seems, Option D might reduce memory usage, it could potentially compromise the evaluation quality by not considering all the necessary metrics. Confused in A/D!

upvoted 1 times

Gudwin 1 year, 6 months ago

Selected Answer: D

D does not harm the evaluation quality.

upvoted 1 times

[Removed] 1 year, 6 months ago

Selected Answer: A

Surely removing evaluation metrics downgrades the quality of the evaluation

upvoted 2 times

frangm23 1 year, 6 months ago

I'm not very sure, but wouldn't be A?.D is degrading evaluation quality (if you're getting less metrics, then the evaluation is worse, at least less complete)

upvoted 2 times

Yajnas_arpohc 1 year, 7 months ago

Selected Answer: A

TFX 0.30 and above adds an interface, with_beam_pipeline_args, for extending the pipeline level beam args per component

tfma.MetricSpec() OOB has recommended metrics; reducing any further might not serve the purpose.

upvoted 2 times

TNT87 1 year, 8 months ago

Selected Answer: D

Add tfma.MetricsSpec () to limit the number of metrics in the evaluation step.

Limiting the number of metrics in the evaluation step using tfma.MetricsSpec() can reduce the memory usage during evaluation and address the out-of-memory error. This can help stabilize the pipeline without downgrading the evaluation quality or incurring additional infrastructure overhead. Running the evaluation step on Dataflow or custom Compute Engine VMs can be resource-intensive and expensive, while migrating the pipeline to Kubeflow would require additional setup and configuration.

ANSWER D

upvoted 4 times

MI06 1 year, 8 months ago

A is wrong , it does not even make sense , the default runner for evaluator component of TFX is data flow so setting runner to dataflow does not change anything , the answer is D because it does not include the any infrastructure minpulation and reduce the memory useable of the TfX component

upvoted 2 times

TNT87 1 year, 8 months ago

https://www.tensorflow.org/tfx/guide/evaluator

upvoted 1 times

TNT87 1 year, 8 months ago

Selected Answer: A

Answer A

upvoted 2 times

TNT87 1 year, 8 months ago

Answer D

upvoted 1 times

RaghavAI 1 year, 9 months ago

Selected Answer: A

https://blog.tensorflow.org/2020/03/tensorflow-extended-tfx-using-apache-beam-large-scale-data-processing.html

upvoted 4 times

Start Learning for free

Social Media

Facebook , Twitter
YouTube , Reddit
Pinterest

We are the biggest and most updated IT certification exam material website.
Using our own resources, we strive to strengthen the IT professionals community for free.