EXAM PROFESSIONAL MACHINE LEARNING ENGINEER TOPIC 1 QUESTION 155 DISCUSSI...

Actual exam question from Google's Professional Machine Learning Engineer
Question #: 155
Topic #: 1
[All Professional Machine Learning Engineer Questions]

You are training an object detection machine learning model on a dataset that consists of three million X-ray images, each roughly 2 GB in size. You are using Vertex AI Training to run a custom training application on a Compute Engine instance with 32-cores, 128 GB of RAM, and 1 NVIDIA P100 GPU. You notice that model training is taking a very long time. You want to decrease training time without sacrificing model performance. What should you do?

A. Increase the instance memory to 512 GB, and increase the batch size.

B. Replace the NVIDIA P100 GPU with a K80 GPU in the training job.

C. Enable early stopping in your Vertex AI Training job.

D. Use the tf.distribute.Strategy API and run a distributed training job.

Show Suggested Answer

by powerby35 at *July 13, 2023, 12:45 p.m.*

Comments

fitri001 Highly Voted 6 months ago

Selected Answer: D

Large Dataset: With millions of images, training on a single machine can be very slow. Distributed training allows you to split the training data and workload across multiple machines, significantly speeding up the process.

Vertex AI Training and tf.distribute: Vertex AI Training supports TensorFlow, and the tf.distribute library provides tools for implementing distributed training strategies. By leveraging this functionality, you can efficiently distribute the training tasks across the available cores and GPU on your Compute Engine instance (32 cores and 1 NVIDIA P100 GPU).

upvoted 5 times

baimus Most Recent 1 month, 1 week ago

Selected Answer: D

Some strategies, like tf.distribute.MirroredStrategy, can provide performance optimizations even on a single GPU. For example, it can take advantage of better gradient computation or data parallelism during backpropagation, which can slightly optimize performance.

upvoted 1 times

Prakzz 3 months, 3 weeks ago

Same Question as 96?

upvoted 1 times

pinimichele01 6 months ago

Selected Answer: D

https://www.tensorflow.org/guide/distributed_training#onedevicestrategy

upvoted 1 times

guilhermebutzke 8 months, 2 weeks ago

Selected Answer: D

D. Use the tf.distribute.Strategy API and run a distributed training job.

Here's why:

A. Increase instance memory and batch size: This might not be helpful. While increasing memory could help with loading more images at once, the main bottleneck here is likely processing these large images. Increasing the batch size can worsen the problem by further straining the GPU's memory.
B. Replace P100 with K80 GPU: A weaker GPU would likely slow down training instead of speeding it up.
C. Enable early stopping: This can save time but might stop training before reaching optimal performance.
D. Use tf.distribute.Strategy: This allows you to distribute the training workload across multiple GPUs or cores within your instance, significantly accelerating training without changing the model itself. This effectively leverages the available hardware efficiently.

upvoted 4 times

bcama 1 year, 1 month ago

Selected Answer: D

perhaps the fact that the second or more GPUs are created is implied and the answer is D
https://codelabs.developers.google.com/vertex_multiworker_training#2

upvoted 1 times

[Removed] 1 year, 3 months ago

Selected Answer: B

The same comment as in Q96. If we look at our training infrastructure, we can see the bottleneck is obviously the GPU, which has 12GB or 16GB memory depending on the model (https://www.leadtek.com/eng/products/ai_hpc(37)/tesla_p100(761)/detail). This means we can afford to have a batch size of only 6-8 images (2GB each) even if we assume the GPU is utilized 100% and model weights take 0 memory. And remember the training size is 3M, which means each epoch will have 375-500K steps even in this unlikely best case.

With 32-cores and 128GB memory, we are able to afford higher batch sizes (e.g., 32), so moving to a K80 GPU that has 24GB of memory will accelerate the training.

A is wrong because we can't afford a larger batch size with the current GPU. D is wrong because you don't have multiple GPUs and your current GPU is saturated. C is a viable option, but it seems less optimal than B.

upvoted 2 times

tavva_prudhvi 1 year, 2 months ago

but using the tf.distribute.Strategy API is not limited to multiple GPU configurations. Although the current setup has only one GPU, you can still use the API to distribute the training across multiple Compute Engine instances, each with its own GPU. By running a distributed training job in this manner, you can effectively decrease the training time without sacrificing model performance.

upvoted 3 times

tavva_prudhvi 11 months, 1 week ago

also, Replacing the NVIDIA P100 GPU with a K80 GPU is not recommended, as the K80 is an older, less powerful GPU compared to the P100. This might actually slow down the training process.

upvoted 1 times

Zemni 1 year, 1 month ago

What you say makes sens for the most part except that K80 GPU has only 12GB of DDR5 memory not 24 , https://cloud.google.com/compute/docs/gpus#nvidia_k80_gpus
So that leaves me with the only viable option which is C.

upvoted 1 times

ciro_li 1 year, 3 months ago

Selected Answer: D

https://www.tensorflow.org/guide/gpu
?

upvoted 1 times

ciro_li 1 year, 2 months ago

I was wrong. It's A.

upvoted 1 times

PST21 1 year, 3 months ago

Selected Answer: D

to decrease training time without sacrificing model performance, the best approach is to use the tf.distribute.Strategy API and run a distributed training job, leveraging the capabilities of the available GPU(s) for parallelized training.

upvoted 1 times

powerby35 1 year, 3 months ago

Selected Answer: A

A
since we just have one gpu, we could not use tf.distribute.Strategy in D

upvoted 1 times

powerby35 1 year, 3 months ago

And C early stopping maybe hurt the performance

upvoted 1 times

TLampr 11 months ago

The increased batch size also can hurt the performance if it is not followed by further optimizations with regards to learning rate for example. If early stopping is applied according to common convention, by stopping when the validation loss starts increasing, it should not hurt the performance. However it is not specified in the answer sadly.

upvoted 1 times

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.