SageMaker Multi-Model vs Multi-Container Endpoints

Which Real-Time Inference Option To Use

[Ram Vegiraju](#)
Published in

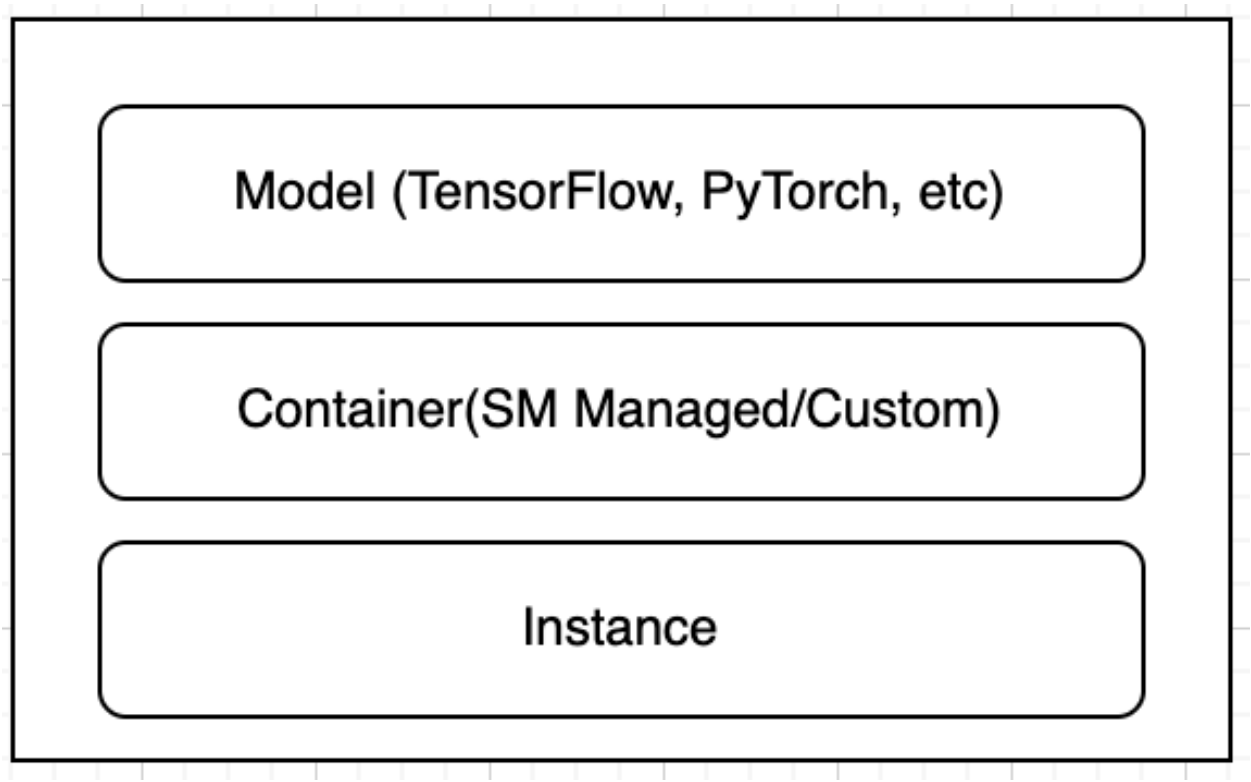
Towards Data Science

.
5 min read
.
Jan 5, 2022
29
1

Amazon SageMaker has a plethora of inference options for model hosting and deployment. Within inference in specific there are four main options:

1. Real Time Inference
2. Serverless Inference
3. Batch Transform
4. Asynchronous Inference

For the purpose of this article we will focus on **Real-Time Inference**. Real-Time Inference is ideal when you need a **persistent endpoint** with stringent latency requirements (sub-millisecond). Within Real-Time Inference there's also different options that you can work with. At the simplest form you can have one endpoint containing one model backed by one instance. Let's take a look at how this basic architecture looks.



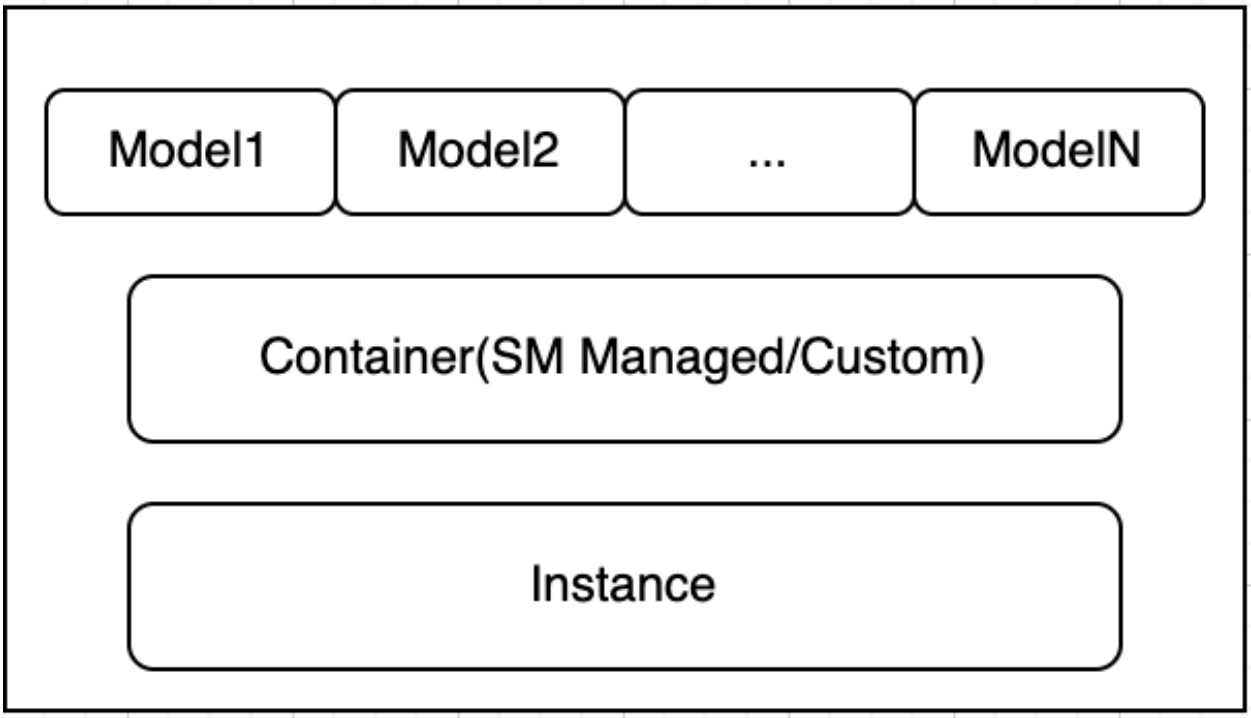
Screenshot by Author

You can create a model and retrieve a SageMaker supported image for popular frameworks such as TensorFlow, PyTorch, Sklearn, etc. If you're working with a custom framework for your model then you can also bring your own container which installs your dependencies.

As your ML platform gets more complex, there's more advanced options such as Multi-Model Endpoints and Multi-Container Endpoints. Let's take a look at the architecture of each to understand their use-cases.

Multi-Model Endpoints (MME)

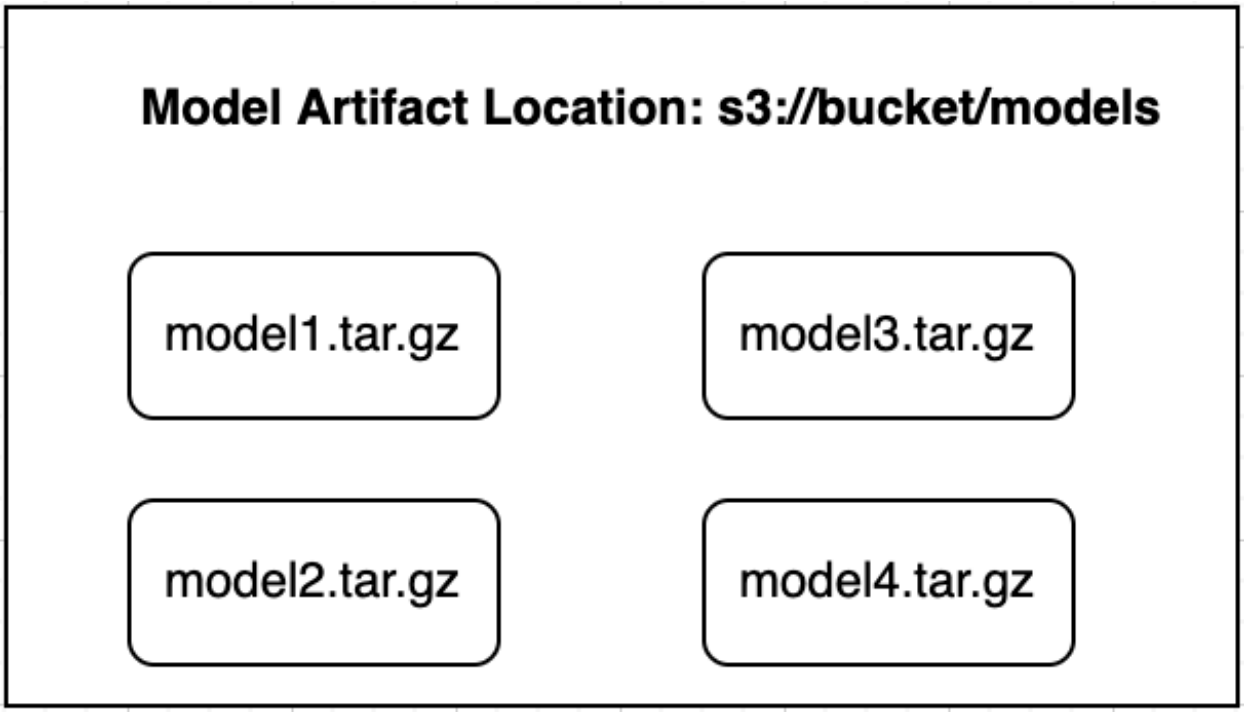
Multi-Model Endpoints help you scale thousands of models into one endpoint. By using a shared serving container, you can host multiple models in a cost-effective scalable manner within the same endpoint. The architecture would differ from a single model endpoint as displayed in the following image.



MME Architecture (Screenshot by Author)

The underlying infrastructure (container + instance) remains the same, but the main difference comes in the models that you can load

up on this container. Similar to a single model endpoint for deployment you need to point to the **model data**. The key difference here is that rather than one, you will have multiple model data/artifacts that you need to load up for each model you have. This artifact like single model endpoints is stored in S3, but for SageMaker to realize you're dealing with a multi-model endpoint you will put all model data into a common S3 location. The model data will also need to be zipped into a modelname.tar.gz format for SageMaker to understand.



Model Data Structure (Screenshot by Author)

The most important feature to understand with MME usage is that **all models must be built in the same framework**. For example, all models in a MME should be purely PyTorch, or purely TensorFlow, or purely a custom framework you are dealing with. **You cannot mix**

and match frameworks for models with a Multi-Model Endpoint.

Now a question might be why not just have a bunch of single model endpoints, rather than loading up all models on a **Multi-Model Endpoint**. Simply put it is much more **cost effective** to have all your models loaded on one endpoint. For instance, say you have 10 models you wish to deploy. If we deploy a **ml.c5.large** instance for each endpoint we have 10 persistent endpoints running. If we look at the pricing for this instance it's $.102 per hour.

10 Single-Model Endpoints: 10 * $.102 = $1.02 per hour

On the other hand if we just have all 10 models loaded up one Multi-Model Endpoint with a ml.c5.large instance we save 10x.

1 Multi-Model Endpoint (10 models): 1 & $.102 = $.102 per hour

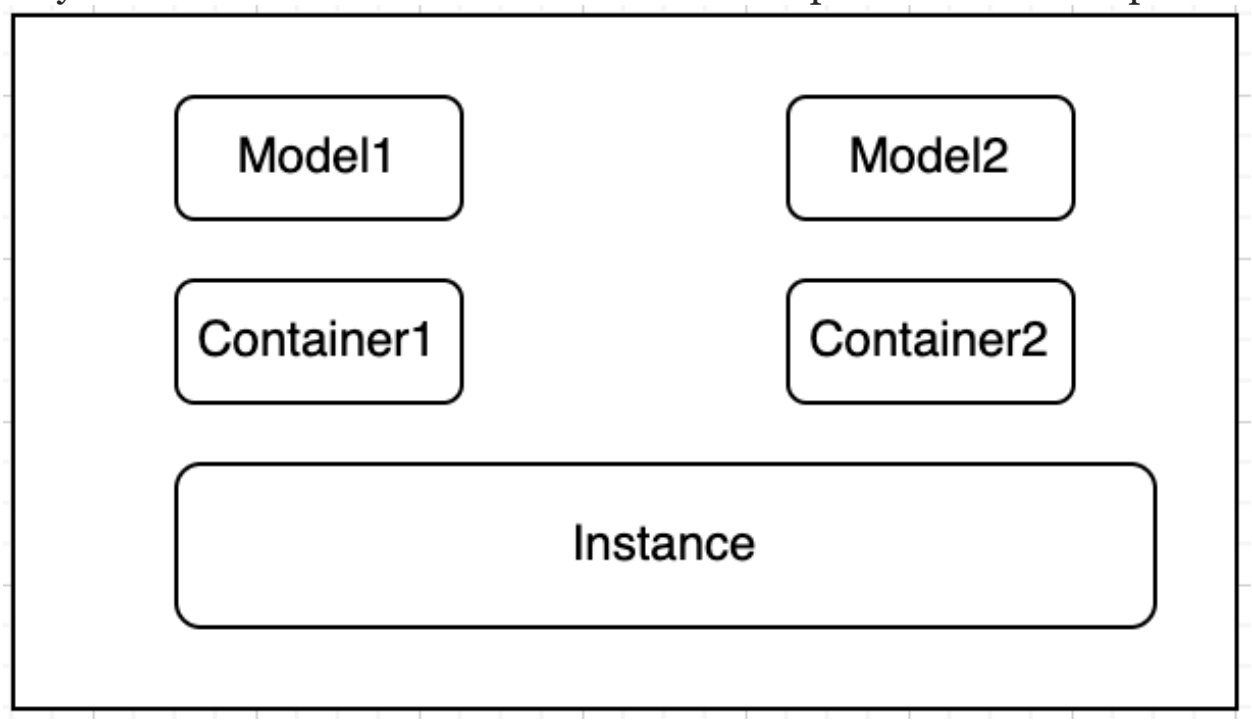
Multi-Model Endpoint Resources

Multi-Model Endpoint TensorFlow Example

Multi-Model Endpoints Documentation

Multi-Container Endpoints (MCE)

The one caveat we discussed with Multi-Model Endpoints is that we cannot mix and match frameworks. Multi-Container Endpoints address this issue, you can provide containers for your different frameworks that you will be working with. For instance you could have a PyTorch and TensorFlow container loaded up on the same endpoint.



Multi-Container Structure (Screenshot by Author)

Multi-Container Endpoints also offer the power in that you can stitch together containers in a Serial Inference Pipeline or invoke the container of your choice. A Serial Inference Pipeline lets you stitch together 2–15 containers, the output of one becomes the input of the next container in sequence essentially. This is an ideal use case for example if you have Sklearn preprocessing container -> TensorFlow Model.

Multi-Container Endpoint Resources

Multi-Container Example

Serial Inference Pipeline Example

Code Difference

Many of the steps working with SageMaker can be done through the Boto3 Python SDK where you can create a SageMaker client. The three main steps of deployment for any SageMaker endpoint are the following.

1. SageMaker Model Creation (Provide model data + image)
2. SageMaker Endpoint Configuration Creation (Take model name, add instance details/config before creating endpoint)
3. Endpoint Creation (Takes 3–4 minutes implements details from Endpoint Configuration)
4. Endpoint Invocation

Like any other real-time endpoints, Multi-Model and Multi-Container Endpoints follow this same recipe. There are minor differences that need to be added for a few of these steps that differentiate the two.

Model Specification

For Multi-Model we want to specify our model prefix (S3 location) that contains our model artifacts. For Multi-Containers we need to specify

our different images/containers that we are dealing with in this use case.

MCE Model Creation

Here we specify our two different containers with the Containers parameter in the create_model call.

The next main difference to be aware of is how the endpoint is invoked and this is pretty self-explanatory through the code samples below.

MME Invocation

For MME you specify a parameter known as TargetModel in the invocation and you specify the model you want to invoke.

MCE Invocation

For MCE similarly you have a parameter known as TargetContainerHostname where you specify the container you want to invoke.

Conclusion & Additional Resources

Amazon SageMaker has an ever-expanding list of options for Inference. It's important to use the option that is most effective for your use-case and I hope this article has clarified which of these two options to use. I'm attaching a few more resources below around Inference and SageMaker as a whole.

SageMaker Inference Examples Repository

SageMaker Inference Recommender To Select The Right Instance

If you enjoyed this article feel free to connect with me on LinkedIn and subscribe to my Medium Newsletter. If you're new to Medium, sign up using my Membership Referral.

Sagemaker