

ADE Deployment new Proposal

- [Overview](#)
 - [Key Changes](#)
 - [Key Advantages](#)
- [Deployment Diagram](#)
- [Deployment resources](#)
- [Steps for deploying a model](#)
- [Steps to tear down an endpoint](#)
- [Model Metadata file](#)
- [Workflow template](#)
- [Post deployment steps](#)
 - [Step 1: Upload error rate file](#)
 - [Step 2: Register with ML Orchestrator](#)
 - [Step 3: Register with Learning Store](#)
- [Addressing Backward compatibility](#)
- [Work Breakdown](#)
 - [Tickets](#)
 - [Final set of tasks](#)

Overview

In order to achieve robust deployment, we are coming up with new pipeline strategy.

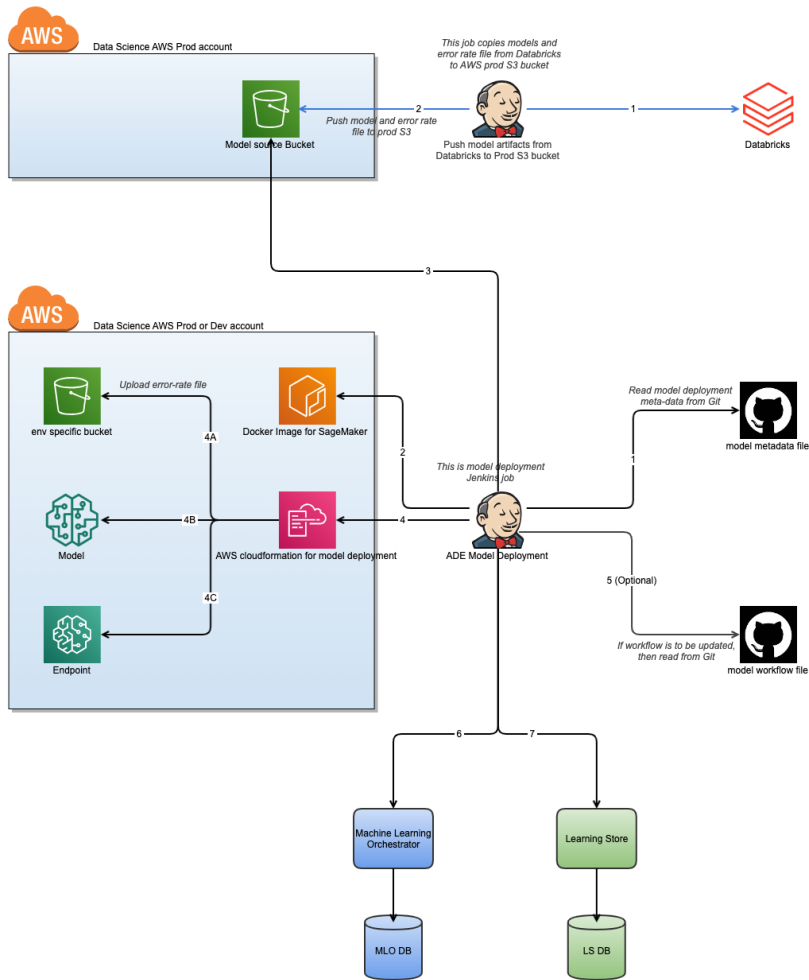
Key Changes

1. **Model Version** - The model version would be same as that on Data bricks. We would use 1-1 mapping here.
2. Using **cloud formation** for deployment - Instead of deploying based on command line, we would use cloudformation template and also use it to maintain state of deployments.
3. **Storing deployment metadata to Git** - Like we started doing now.
4. Using **one endpoint for different iterations of a model** - Instead of deleting and creating new sagemaker endpoints, we will use one endpoint for multiple versions of a model.
We will deploy new endpoint when we have significant change and we would want to support two version of the same.
5. **Multiple model versions for same endpoint** - Extending on the point above, we would create a model and deploy on same endpoint and rollback when needed.
6. **Template based workflow** - Workflow would be saved as a configuration on Git, we would use template engine to replace data at the time of registration of workflow.

Key Advantages

1. Reduce model version confusion by keeping 1-1 version number with DataBricks.
NOTE: It is expected that ML engineer would know which version is in production.
2. Reduce update to model registration. It only updates when endpoint changes.
3. Model deployment is independent of release version. One model can serve multiple release versions.
4. Workflow can be designed by ML engineer without requiring any change on deployment (mostly)

Deployment Diagram



Deployment resources

Model Artifacts pushed from Databricks

Resource	Naming strategy	Notes	example
ADE Artifacts S3 Bucket	em-ds-prod-model-artifacts	Same as today	
ADE Artifacts folder	em-ds-prod-model-artifacts/document-extraction/v2<model_name>/<version>/	new folder structure	em-ds-prod-model-artifacts/document-extraction/v2/w2llm/11/
ADE Artifact - model	em-ds-prod-model-artifacts/document-extraction/v2<model_name>/<version>/model.tar.gz		em-ds-prod-model-artifacts/document-extraction/v2/w2llm/11/model.tar.gz
ADE Artifact - error rate	em-ds-prod-model-artifacts/document-extraction/v2<model_name>/<version>/error_rate.json		em-ds-prod-model-artifacts/document-extraction/v2/w2llm/11/error_rate.json

Deployment Resources created by ML Engineer for deployment

Resource	Naming strategy	Notes	example
deployment metadata	deployment/<model>.yaml	There will be only one version, the one which will be in production	deployment/w2llm.yaml
teardown-metadata	teardown/<model>.yaml	If a model version is to be torn down, then it should be moved to teardown folder.	teardown/w2llm.yaml
Workflow-template	<model>/<model>-<version>.json	Version can be incremental decided by ML Engineer. This has to be updated in deployment metadata file. Refer this	w2/w2llm-01.json

AWS resources created by new deployment job

Resource	Naming strategy	Notes	example
SageMaker model	ml-ade-<environment>-<model_name>-model-<version>	New model name	ml-ade-dev-w2llm-model-11
SageMaker endpoint	ml-ade-<environment>-<model_name>-model-endpoint-<sagemaker-version>	sagemaker version is referred from deployment metadata file	ml-ade-dev-w2llm-model-endpoint-1
SageMaker endpoint config			

NOTE: We will discontinue using deployment metadata file push from databricks.

MLO database resource is discussed [under this section](#).

Steps for deploying a model

To deploy a new model, you would need to do the following:

1. Create your model and make sure it is tested against the latest error rate file.
NOTE: Error rate file could be automatically updated by LS
2. Create/Update model deployment meta data file and check in to Git
 - a. This file needs to go through PR process.
 - b. It should be possible to deploy from a branch.
3. OPTIONAL - Tear down an existing endpoint.
4. Create workflow template
 - a. There can be multiple versions of this file.
 - b. A version which is on prod cannot and should not be overwritten.
5. Deploy artifacts from Databricks to AWS S3.
 - a. For new models, use the new job.
6. Deploy model to AWS environment using new Jenkins job. Specify the following params:
 - a. Environment
 - b. Branch
 - c. Force deploy workflow ?

Steps to tear down an endpoint

To begin with, there is no way to tear down a model in current phase, you can only tear down endpoints. For 22.2 release tear down is needed because ML engineers should have flexibility to test and retest with confidence. The 22.2 release is the first time so all models would be deployed first time. We need to ensure that ML engineers can test a fresh deployment so in case they have deployed something they should be able to tear and then deploy which should be like a new deployment.

1. Run the new tear dow Jenkins job.

This job would

1. Delete entry from MLO database - **DO WE HAVE AN API FOR THIS?**
2. Delete AWS Sagemaker endpoint model
3. Delete AWS Sagemaker endpoint-config model
4. Delete AWS Sagemaker model

Model Metadata file

Metadata file will be checked into git. The file itself will not have version, but the branch would represent the version going to production. We can plan to have a master deployment file that could be used to deploy multiple jobs in one go.

		Notes
Default Branch	master	
File format	YML	We can finalize between JSON and YAML

Versioning	Not Applicable	No versioning.
Master deployment		

Git Project structure

```

- deployment
  - w2llm.yaml
  - w2copydetector.yaml
  - ...
- tear down
  - paystub.yaml
  - ...
- workflow
  - w2
    - w2llm-01.json
    - w2llm-02.json
  - PayStub
    -paystub-01.json
- bulk-deployment
  - release-22-1.json
- test
  ... any test we can plan to validate json or YAML.
```

w2llm

```

model:
  name: w2llm
  workflow: w2llm-01.json
  inference-type: Value Extractor
  provider: SageMaker
  version: 12          //This is model version. This is how deployment will search articles during deployment
from S3
  model-image-name: databricks-ml-73lts-cpu
SageMaker:
  version: 01          // Cloudformation will use this to deploy new or update existing.
scaling:
  dev:
    instace-type: ml.c4.xlarge
    autosacling: false
  peg:
    instace-type: ml.c4.xlarge
    autosacling: true
    min-instances: 3
    max-instances: 6
    scaling: #Currently, only single scaling option is available.
      - name: CPUBasedScaling
        type: CPUUtilization
        stat: Average
        Percent: 70
        scaling-out-period: 30
        scaling-in-period: 300
  scheduled-scaling:
    start-time-cron: 0 9 * * ? *
    min-capacity: 1
```

Workflow template

The template would be designed by ML engineer and checked into git. This workflow would be referred in deployment metadata file. Check this section above for details.

Code will use [mustache template engine](#) to replace values in template.

Pending: See if we can pull these values from AWS cloudformation template? If we can search output, this would be better.

w2llm-01.json

```
{
  "sectionNames": [],
  "classNames": "IRS W-2",
  "invocationSequenceConfig": {
    "staticParametersConfig": {
      "errorRatePath": "{{w2CopyDetector-errorPath}}"
    },
    "extraNoPiiSanitizedMetrics": ["detectedCopies"],
    "sequence": [{
      "modelEndpoint": "mlo-{{env}}-ade-{{w2CopyDetector-endpoint}}",
      "inputParameters": ["normalizedPngImageBytes"],
      "outputParameters": ["detectedCopies"]
    }, {
      "modelEndpoint": "mlo-{{env}}-ade-{{w2Llm-endpoint}}",
      "inputParameters": ["className", "sectionName", "commonOcr", "errorRatePath", "detectedCopies"],
      "outputParameters": ["finalPredictions"]
    }]
  }
}
```

Post deployment steps

Step 1: Upload error rate file

Fetch error_rate file from source S3 and upload to Target S3

Step 2: Register with ML Orchestrator

Registering workflow with MLO

Changes in MLO DB

Stretch goal for 22.2

Step 3: Register with Learning Store

Use the existing approach to register with learning store.

Addressing Backward compatibility


Work Breakdown

EPIC:

DS-6025 - Getting issue details...

STATUS

Tickets

key	summary	type	created	updated	due	assignee	reporter	priority	status	resolution
<div><div> Jira project doesn't exist or you don't have permission to view it.</div><div>View these issues in Jira</div></div>										

Final set of tasks

TODO: Move this under epics and refer epic here.

	Task	Jira	Complexity (1-5)
1	Create new Jenkins job to push artifacts from Databricks to S3		
2	Create new repo for new model deployment		1
3	Create code for reading data from model meta data to trigger Cloudformation		2
4	Create cloudformation template for model deployment and sagemaker endpoint		4
5	Spike - Check if cloudformation output be used for generating workflow file		
6	Use mustache template engine to replace workflow data with correct values.		
7	Create Jenkins job to deploy models		1
8	Code to register model with MLO		2
9	Code to register model with LS		
10	MLO changes for model inference table and API changes		3