Testing Cloud-to-Edge Deep Learning Pipelines: Ensuring Robustness and Efficiency

Rustem Feyzkhanov

Senior Staff Machine Learning Engineer

Instrumental

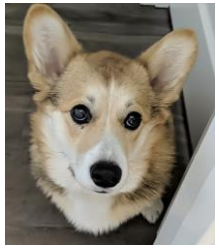
INSTRUMENTAL

Agenda

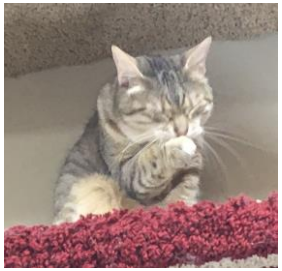
- Introduction to cloud-to-edge deep learning pipelines
- Testing strategies for cloud-to-edge pipelines
- ML pipeline dev tests
- ML pipeline internal tests
- Stage tests
- Summary

Example – corgi/not corgi model

Business understanding → Data acquisition → Modeling → Deployment → Customer acceptance



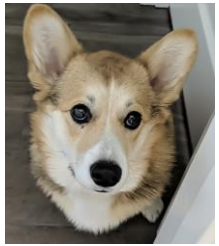
→ Model → corgi



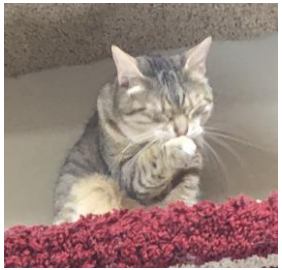
→ Model → not corgi

Example – corgi/not corgi model

Business understanding → **Data acquisition** → Modeling → Deployment → Customer acceptance

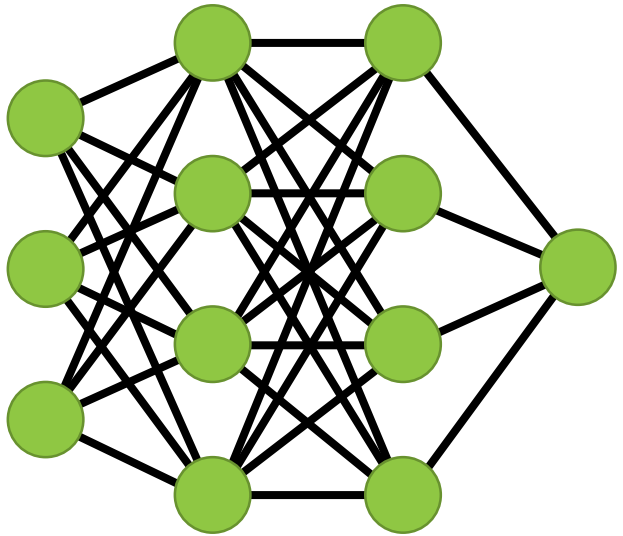






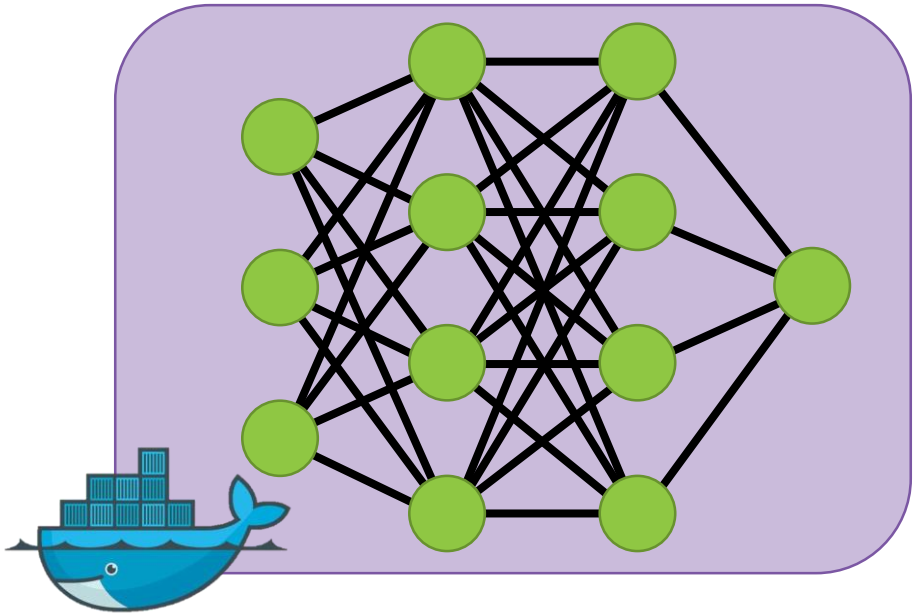

Example – corgi/not corgi model

Business understanding → Data acquisition → **Modeling** → Deployment → Customer acceptance



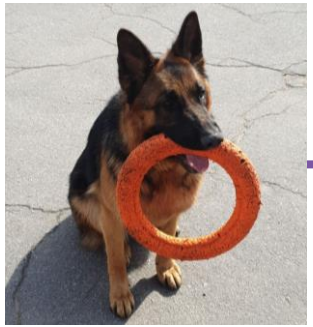
Example – corgi/not corgi model

Business understanding → Data acquisition → Modeling → **Deployment** → Customer acceptance



Example – corgi/not corgi model

Business understanding → Data acquisition → Modeling → Deployment → Customer acceptance



Prediction service ...

Why did it happen?

corgi ← Prediction service ...

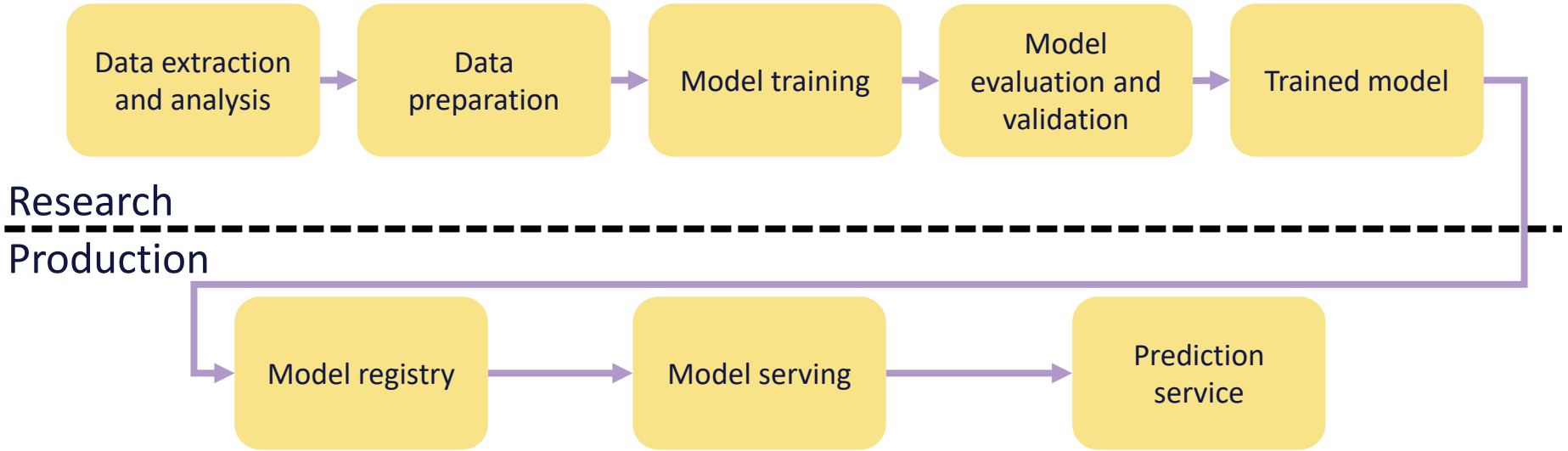
Possible edge cases with model on edge

- Data preprocessing is different between cloud and edge
- Inference framework has a different version and doesn’t support model
- NPU drivers are not backward compatible and drop some of the tensors
- Data drift happened on edge
- Training set wasn’t comprehensive enough

=> proper testing can solve most of the above issues (not all though)

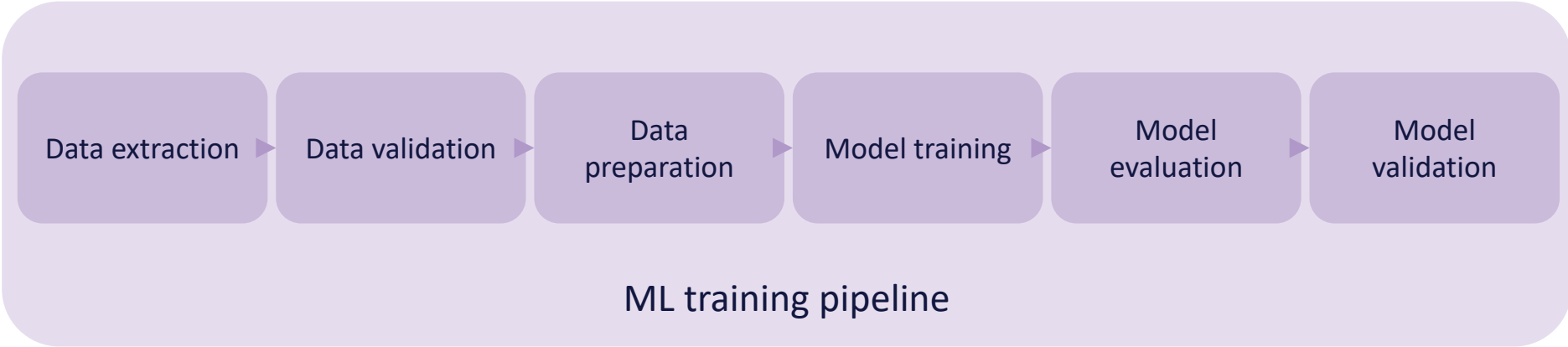
Cloud-to-edge deep learning pipeline evolution

Cloud-to-edge deep learning pipeline – phase 1



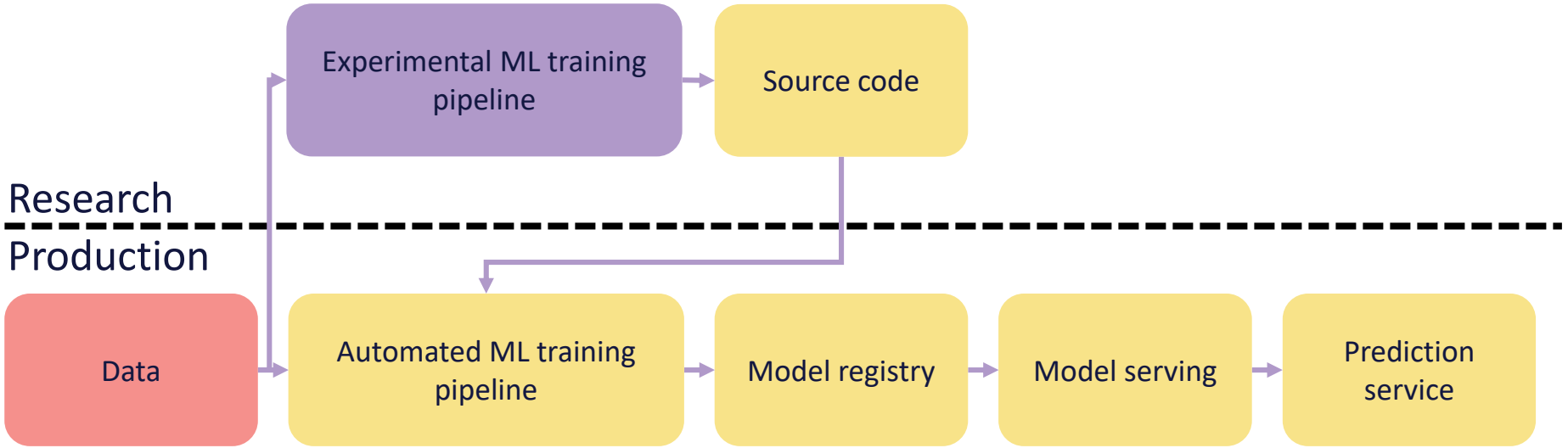
From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Cloud-to-edge deep learning pipeline – phase 2



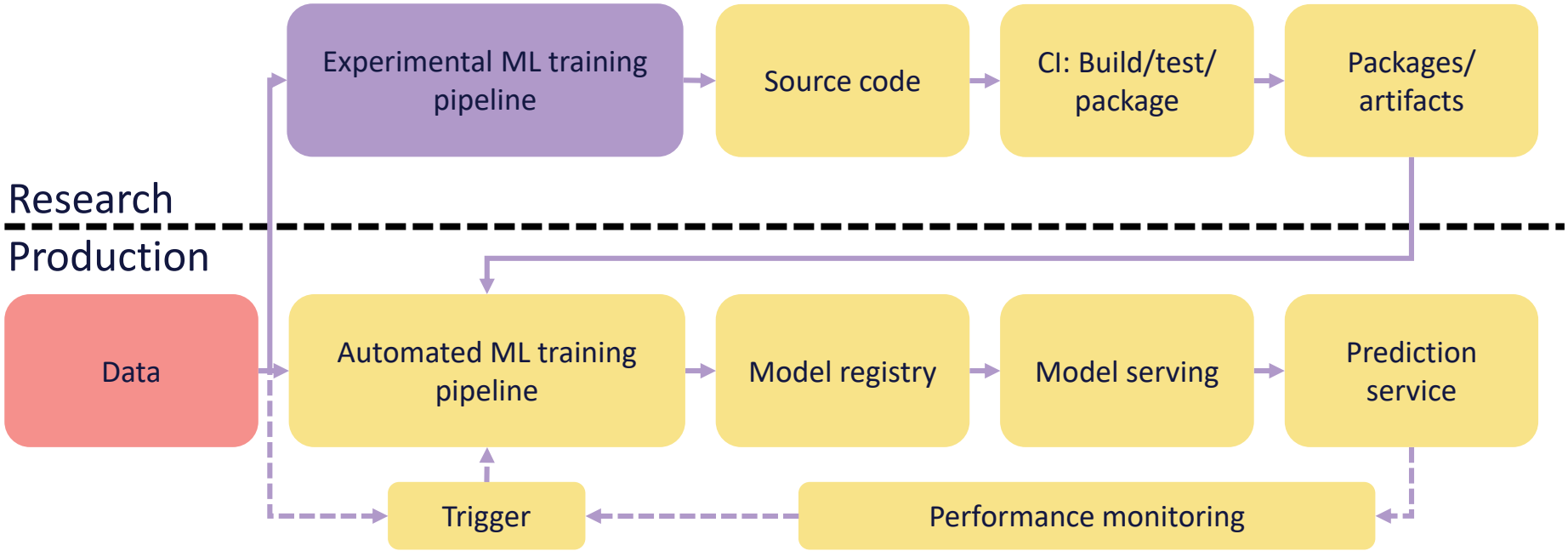
From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Cloud-to-edge deep learning pipeline – phase 2



From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Cloud-to-edge deep learning pipeline – phase 3



From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Cloud-to-edge deep learning pipeline – phase 3

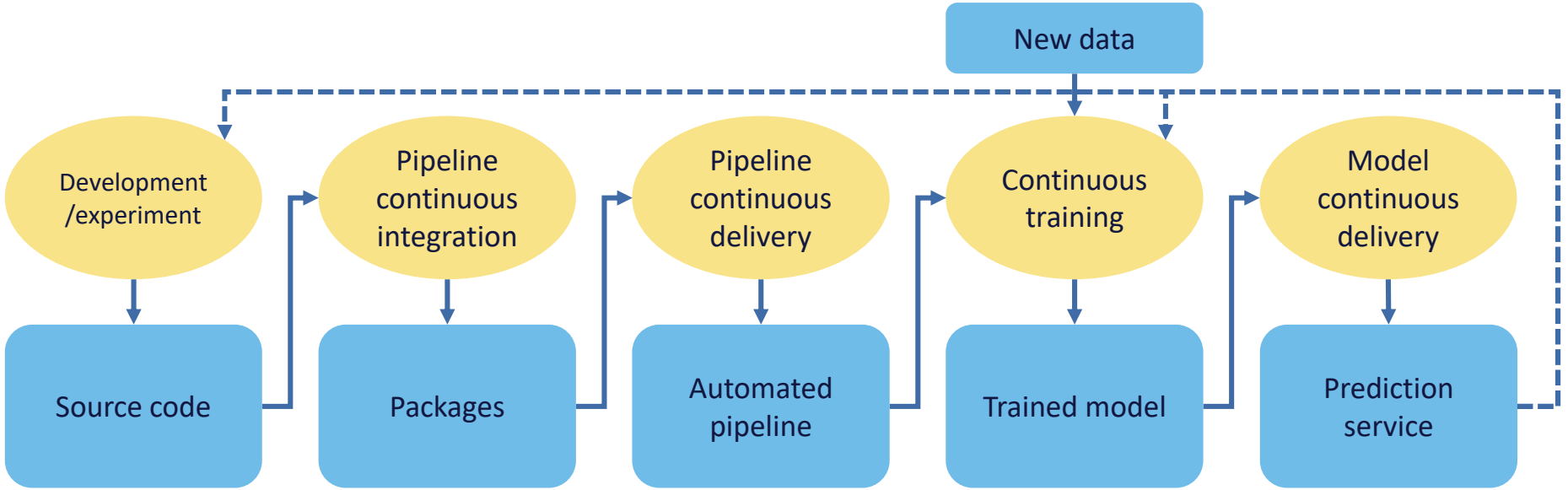
Phase 3

- Pros
  - Fully automated model training
  - Quick releases/model retraining
- Cons
  - New model may run into errors on the edge device
  - New model may perform on the edge differently from the cloud

Cloud-to-edge deep learning pipeline testing

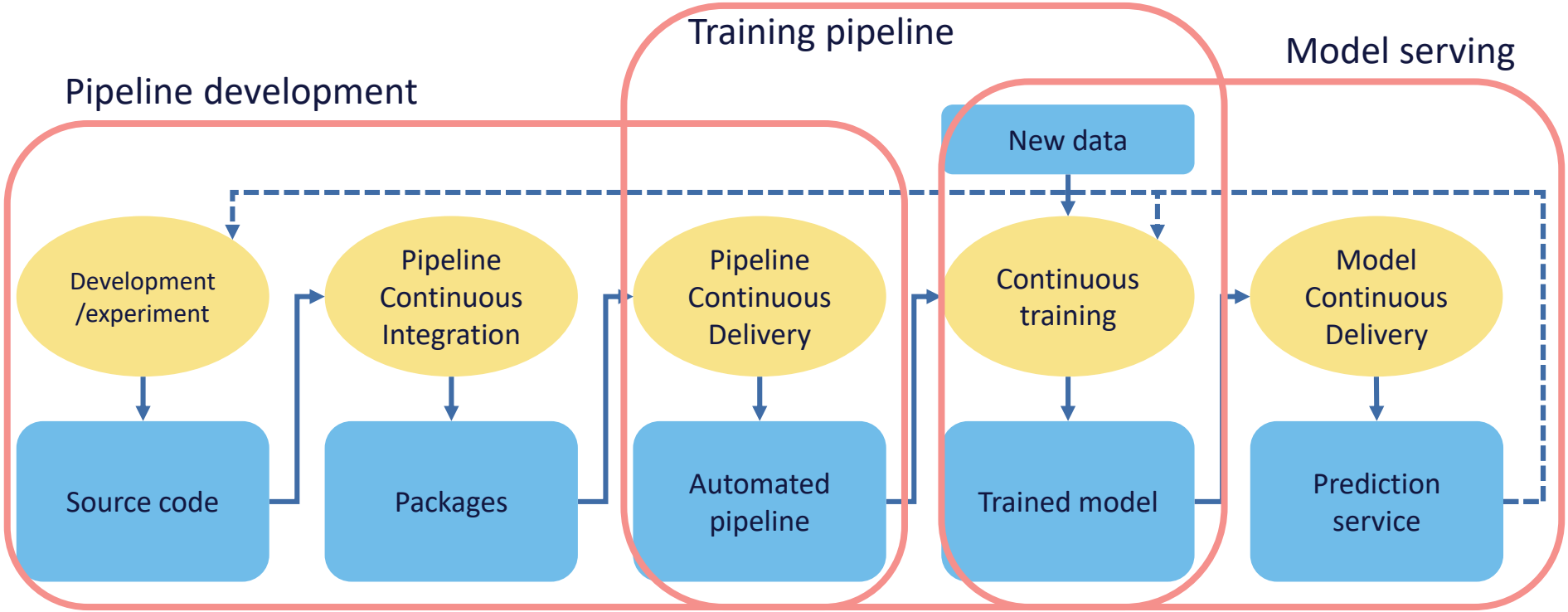
Cloud-to-edge deep learning pipeline – model lifecycle

Let’s unwrap the pipeline into ML model lifecycle



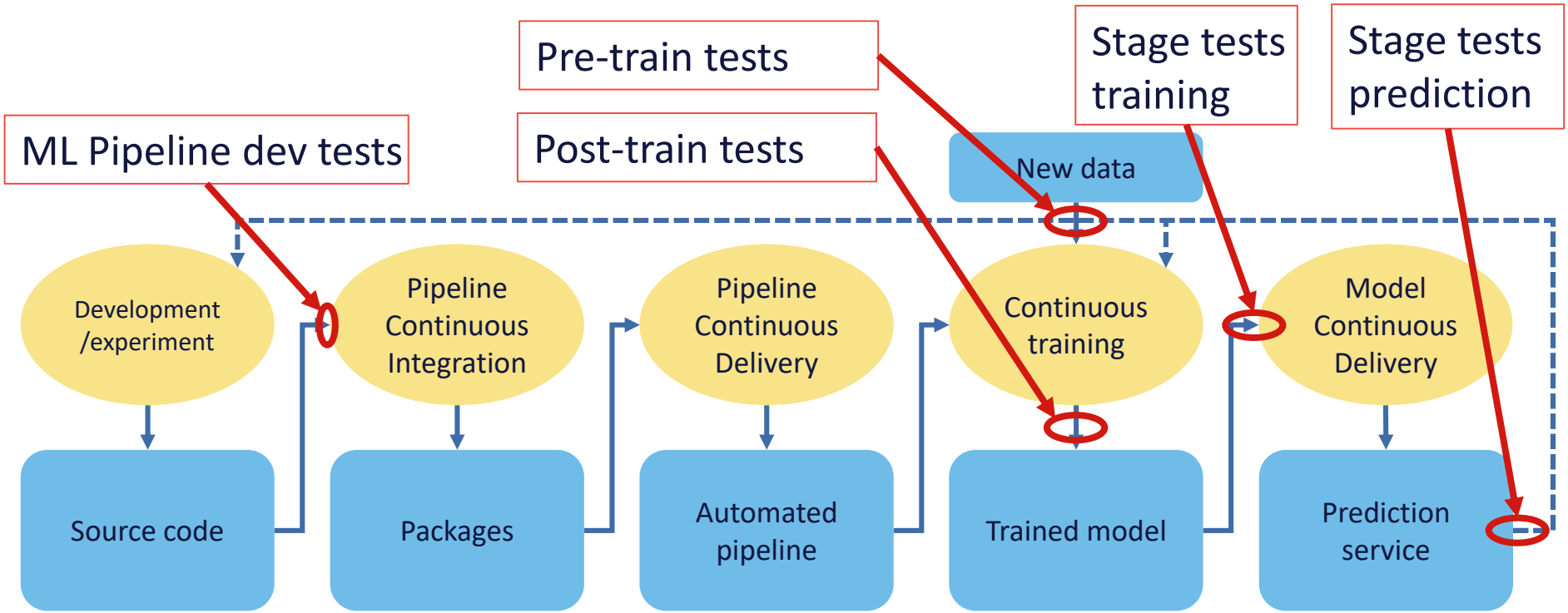
From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Cloud-to-edge deep learning pipeline – model lifecycle



From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Cloud-to-edge deep learning pipeline – model lifecycle



From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Categories of tests

ML pipeline tests	Pre-train tests	Post-train tests	Stage test – training	Stage test – prediction
Run in local dev and CI	Run before training in the cloud	Run after training in the cloud	Run in stage env after the training package is deployed	Run in stage env after the inference package is deployed
Test pipeline code before packages are generated	Checks our assumptions about data	Checks our assumptions about model	Checks for regression in the cloud	Checks for regression on edge

ML pipeline tests

ML pipeline tests

Unit tests

Check individual components of the pipeline

Regression tests

Check that change didn't affect existing functionality and artifacts

Smoke tests

Cheap tests to check that model performs as expected on simple dataset

Integration tests

Check that components work together in an expected way

ML pipeline tests examples

Unit tests

- Data split/preparation
- Naive cases for training
- Training-serving skew

Regression tests

- Backward compatibility with old models
- Data shift due to preprocessing library change

Smoke tests

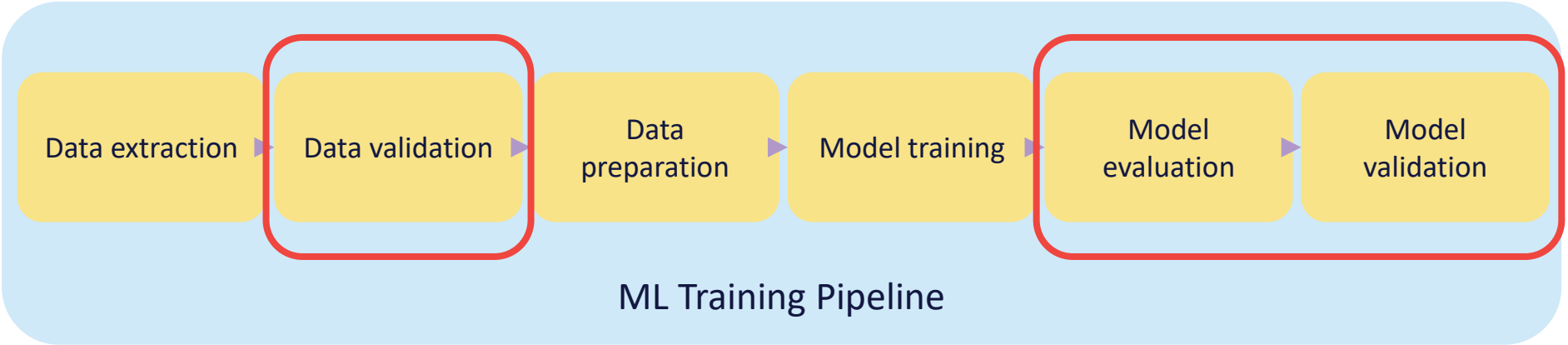
- Does model correctly predict sample from train?
- Does model return result in expected format?

Integration tests

- Test that pipeline works together
- Test that there is no discrepancy between train and serving env

ML pipeline internal tests

Cloud-to-edge deep learning pipeline



From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Pre-train and post-train tests

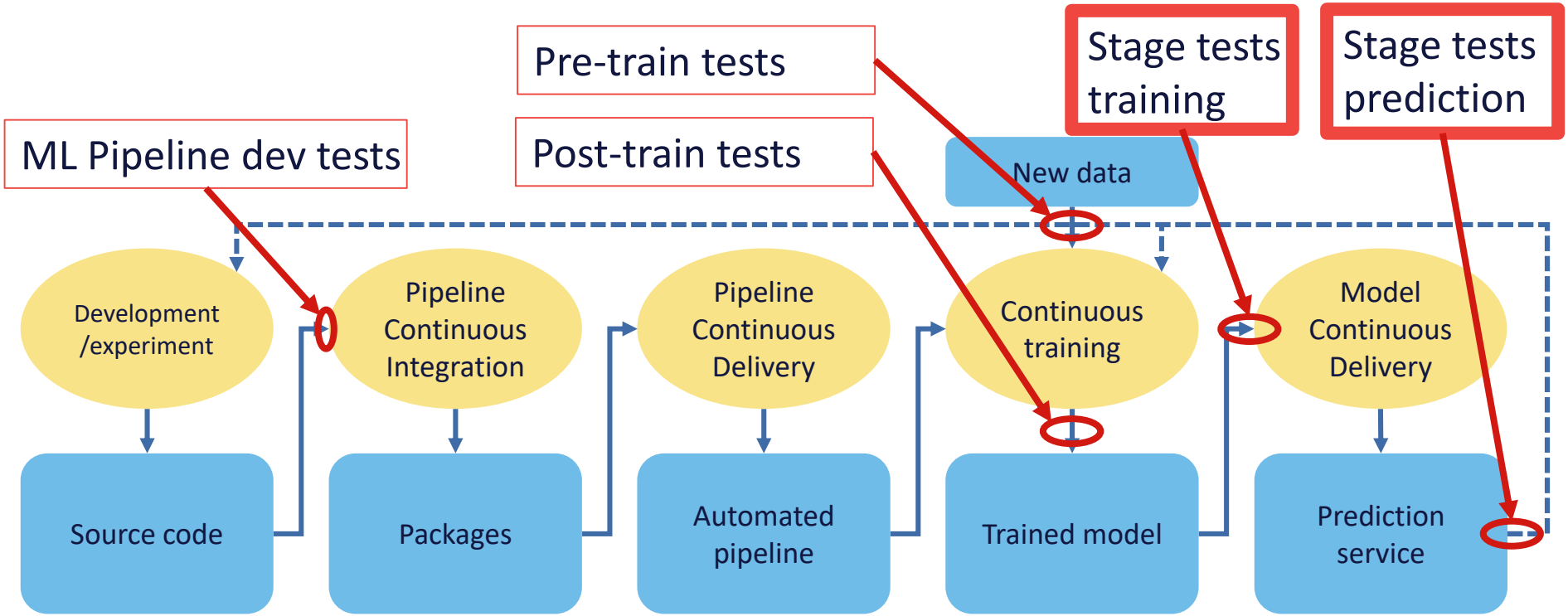
Pre-train tests	Post-train tests			
Data validation	Model evaluation	Model validation		
Explicit checks for data which we will use for training	Performance on a validation or test dataset	Invariance tests	Directional expectation tests	Minimum functionality tests
Tests that identify data/pipeline assertion failure and fail early	Tests that model was able to converge and reach acceptable performance	Test that specific data perturbations don't affect model output	Test that specific data perturbations do affect model output	Ensure that model works in critical scenarios/failure modes/edge cases

Pre-train and post-train tests

Pre-train tests	Post-train tests			
Data validation	Model evaluation	Model validation		
Explicit checks for data which we will use for training	Performance on a validation or test dataset	Invariance tests	Directional expectation tests	Minimum functionality tests
- Incorrect labels - Incorrect features	- Low accuracy	- Rotated/shifted image have similar predictions	- Apples are at the top when searching for apples	- Blurry images - Overexposed images

Stage tests

How cloud-to-edge deep learning pipeline looks like



From https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Stage tests

- Run on stage (could be dev environment)
- Tests should expose bugs with more realistic data and load
- Tests could be more complex
- Examples
  - Shift in prediction scores
  - Shift in predictions themselves
  - CPU/RAM utilization (to catch memory leak)

Summary

Summary

- Tests are a way to ensure predictable and transparent model behavior
- Tests are the best way to catch ML bugs early
- ML tests ≠ classic software engineering tests, but similar mindset could be applied:
  - Unit tests to check for bugs
  - Testing specific scenarios
  - Testing expected edge cases

Resources

- Posts and presentations
  - https://arseny.info/reliable_ML
  - https://www.jeremyjordan.me/testing-ml/
  - https://eugeneyan.com/writing/testing-ml/
  - https://krokotsch.eu/cleancode/2020/08/11/Unit-Tests-for-Deep-Learning.html
  - https://cloud.google.com/solutions/machine-learning/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning
- Github Repositories
  - https://github.com/marcotcr/checklist
  - https://github.com/great-expectations/great_expectations
  - https://github.com/HypothesisWorks/hypothesis