


Tecton Enrichment Handler for Beam

Proof of Concept: *#36062*
Tracking Beam Issue: *#35046*
Last updated: *Sep 12, 2025*
Authors: *Mohamed Awnallah*
Status: ***Draft*** *> Current > Needs update > Obsolete*

Overview

The Tecton Feature Store Enrichment Handler for Apache Beam enables streaming and batch pipelines to enrich data with features from Tecton's online feature stores. This component seamlessly integrates Tecton's capabilities into Beam data processing workflows, allowing pipelines to retrieve pre-computed or real-time features for various machine learning use cases.

Objective

Build a configurable and efficient Beam Enrichment Handler for Tecton that supports enriching data with features from Tecton's online feature stores. It should enable seamless integration of feature retrieval with Beam pipelines for real-time serving.

Background

Beam currently has Vertex AI and Feast feature store enrichment handlers [1] [2], which enable pipelines to retrieve features for enrichment. By integrating Tecton Feature Store enrichment

handler with Beam, this will enable existing Tecton users to leverage Beam's powerful data processing capabilities for their feature engineering ML pipelines, creating a natural pathway for adopting Beam in feature engineering ML pipelines.

Design and Implementation

The Tecton Feature Store Enrichment Handler for Beam is designed to efficiently retrieve features from Tecton's online feature store. This component will leverage Tecton's client SDKs to interact with the online feature store APIs, enabling Beam pipelines to enrich incoming data with real-time features. The handler will transform Beam elements to and from a format compatible with Tecton's online feature serving APIs and batch queries to optimize performance.

Constraints

- Online Feature Retrieval Scope: The Tecton Feature Store Enrichment Handler for Beam is currently limited to retrieving features solely from Tecton's online feature store.

Client SDK Selection

Tecton provides two Client SDKs: tecton-cli (lightweight) [3] and tecton (heavy-weight) [4]. For online feature retrieval, tecton-cli is sufficient and provides the necessary functionality. In case of tecton-cli python package to be added as extra dependency for alignment with Beam vision as seen in Beam 3.0.0 Milestone 1 ("Consider reducing dependencies in core, moving more to separate/extra packages") [5].

Connection Parameters

The Tecton Feature Store Enrichment Handler for Beam defines its configuration through the TectonConnectionConfig data class, which encapsulates all necessary parameters for connecting to Tecton's online features store. This configuration is passed to the TectonFeatureStoreEnrichmentHandler.Here's the updated "Enrichment Parameters" section based on the provided TectonFeaturesRetrievalConfig data class.

Parameter	Type	Description	Default
url	str	The URL of the Tecton instance to connect to.	*Required*

default_workspace_name	str	The name of the workspace containing the feature service.	*None*
api_key	str	The API key for authenticating with the Tecton instance.	*Required*
kwargs	Dict\[str...	Additional keyword arguments for connection operations. Enables forward compatibility.	None

Enrichment Parameters

The Tecton Feature Store Enrichment Handler for Beam defines its configuration through the TectonFeaturesRetrievalConfig data class, which encapsulates all necessary parameters for retrieving features from a Tecton feature store. This configuration is passed to the TectonFeatureStoreEnrichmentHandler.

Parameter	Type	Description	Default
feature_service_name	str	The name of the feature service containing the features to fetch from the online Tecton feature store. This should match a feature service defined in your Tecton workspace.	*Required*

entity_id	str	The entity name for the entity associated with the features. The entity_id is used to extract the entity value from the input row. Please provide exactly one of entity_id or entity_row_fn.	"" (Empty St...
entity_row_fn	Optional[Ent...	A lambda function that takes an input beam.Row and returns a dictionary with a mapping from the entity key column name to entity key value. It is used to build/extract the entity dict for feature retrieval. Please provide exactly one of entity_id or entity_row_fn.	None
request_context_map	Optional[Ma...	Optional mapping of request context parameters to pass to Tecton for feature computation. These are typically used for real-time features that depend on request-time data.	None
workspace_name	Optional[str]	Optional workspace name override. If not provided, uses the workspace from the connection config.	None

allow_partial_results	bool	Whether to allow partial results if some features fail to compute.	False
request_options	Optional[Re...	Optional RequestOptions for controlling request behavior.	None
metadata_options	Optional[Me...	Optional MetadataOptions for controlling what metadata is returned.	MetadataO...
kwargs	Dict[str, Any]	Additional keyword arguments for feature retrieval. Enables forward compatibility with future Tecton feature retrieval parameters.	{} (Empty Di...

Metrics

The Tecton Enrichment Handler will expose several metrics using Beam's metrics API, providing insights into its performance and operational health. These metrics will be accessible through Beam's monitoring interfaces (e.g., Dataflow Monitoring UI, Flink UI).

Metric	Description
features_retrieved	A counter that tracks the total number of feature sets successfully retrieved from Tecton, indicating the volume of successful enrichments.
feature_retrieval_requests	A counter that increments each time a request for features is sent to the Tecton online feature store, monitoring API call frequency.

feature_retrieval_errors	A counter that tracks the number of failed feature retrieval requests, crucial for identifying and troubleshooting issues with feature fetching.

Testing Framework

Infrastructure

For development, a free tier version on explore.tecton.ai can be used to test and evaluate the Enrichment Handler.

For deployment, a GitHub secret can be used to store the Tecton API key so that integration tests can be run against a live Tecton instance in the CI environment.

Unit Testing

The unit tests for the Tecton Enrichment Handler focus on validating user-provided data and verifying the conversion logic from Beam row elements to a format compatible with Tecton. These tests are isolated and do not require a running Tecton instance, making them fast, lightweight, and suitable for both local development and continuous integration environments.

Integration Testing

Integration tests validate the behavior of the Tecton Enrichment Handler when interacting with a live Tecton instance under various real-world conditions. These tests are designed to align with Beam I/O integration testing standards [5].

Category	Scenario	Expected Outcome
Feature Retrieval	Feature Service not found	Error (404)
	Workspace not found	Error (404)
	Invalid API Key	Error (401/403)
	Entity ID not found for a given request	Returns row with empty feature values

	Feature Service exists and features available	Success (200), returns enriched data
	Batching behavior under configured thresholds	Records are grouped, queries, and batched responses returned

Notebook Example

A Jupyter notebook example showcasing Tecton Enrichment Handler is planned to be included as part of the official Beam documentation [16]. This notebook will demonstrate key usage patterns, configuration options, and end-to-end data ingestion workflows using the Tecton connector.

It is typically included as a follow-up pull request (PR) to the Enrichment Handler to ensure the documentation remains functional and aligned with the latest code.

Documentation && Website Updates

To reflect the addition of Tecton Enrichment Handler, updating the release notes of the current *unreleased* Beam version to signal that Beam now includes built-in support for Beam as an enrichment handler.

In a follow-up PR, including a Python example and updating the Beam website to list Tecton I/O as an officially supported built-in enrichment handler.

References and Resources

[1] Vertex AI Enrichment Handler | Apache Beam Source Code

[2] Feast Enrichment Handler | Apache Beam Source Code

[3] Tecton Client Package | Python Package Index

[4] Tecton Package | Python Package Index

[5] Beam 3.0.0 Milestone 1 | GitHub Project Milestones

[6] I/O Standards | Apache Beam Documentation