Requirements and Solutions Document for Leveraging Embeddings with NVTabular Workflows for Training and Deployment

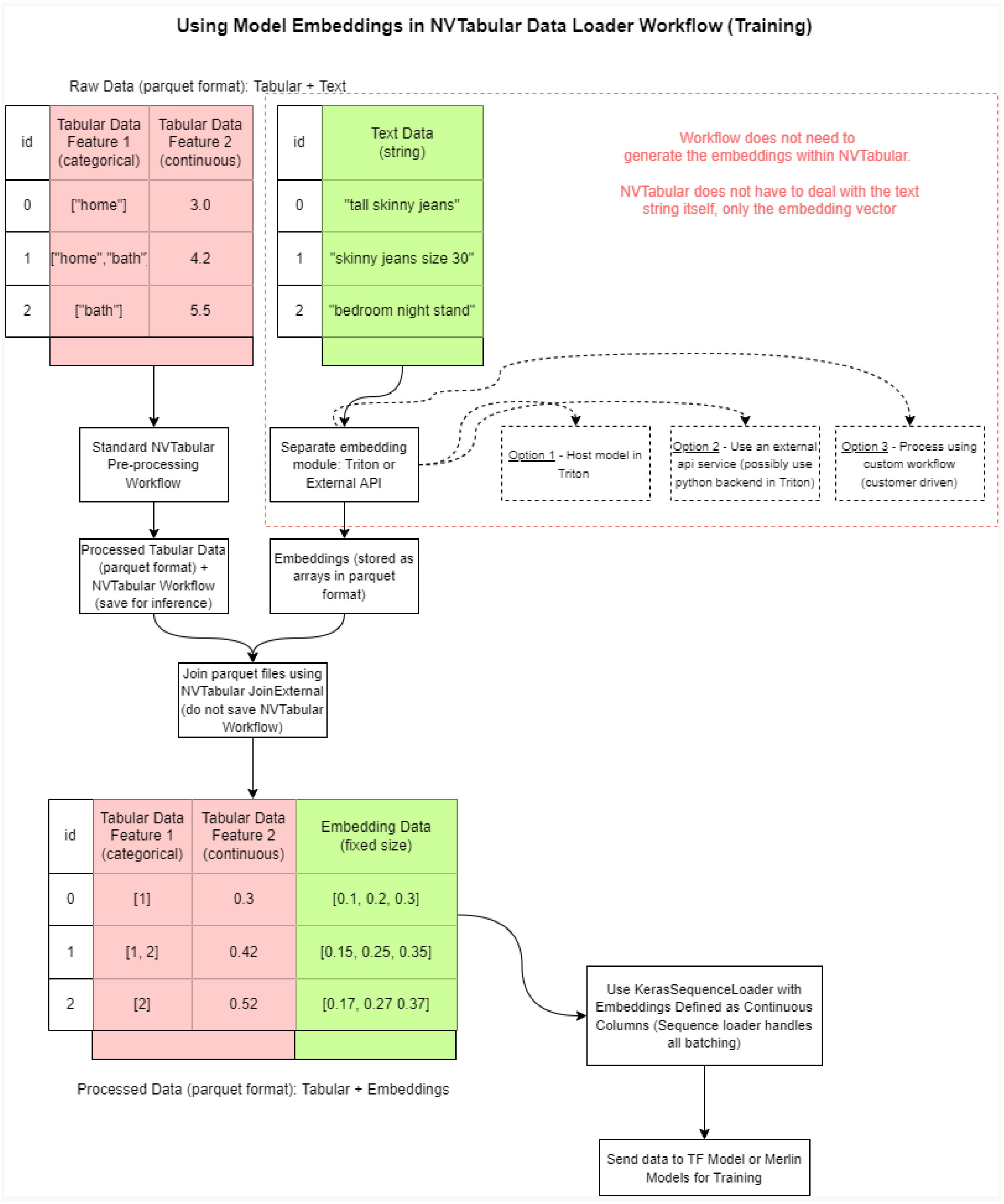
Objective

- Leverage dense vectors/embeddings as model inputs that were obtained from an encoder model (text, image, other) for raw input data (text, image, etc) and leverage these embeddings in NVTabular data loaders during training and inference workloads for RecSys models (including Transformers4Rec)
- Benefits
    - model based embeddings can be generated for all possible entities, which might not be in the training set.
    - model based embeddings can be trained on one task and used on another task, with or without further tuning.
    - With the embeddings as an input feature vector, the number of parameters are reduced compared with learning the embeddings during task training.
    - Workflow may be used in general RecSys models (ideally Merlin Models) along with Transformers4Rec (some alteration may be required)
    - Leverage the speed of GPU processing to complete this workload in fastest time possible

Assumptions

- All input data (of Tabular format) during training should be fetched by the NVTabular Data Loader to optimize training times (through faster data loading)
    - If tabular data is fetched by NVTabular and other data is fetched by the underlying framework (FW - Tensorflow or Pytorch), then coordinating batches of data from two different fetches can be complicated and potentially reduce data loading speeds
- NVTabular as a feature engineering module is not able to host a model (or make a call to an external model service) in its current state
- NVTabular does not have to process the raw input string, just be able to work with the embedding inputs that come from some other text encoding process.
- The workflow for training can be different to the workflow for inference

Using Model Embeddings in NVTabular Data Loader Workflow (Training)

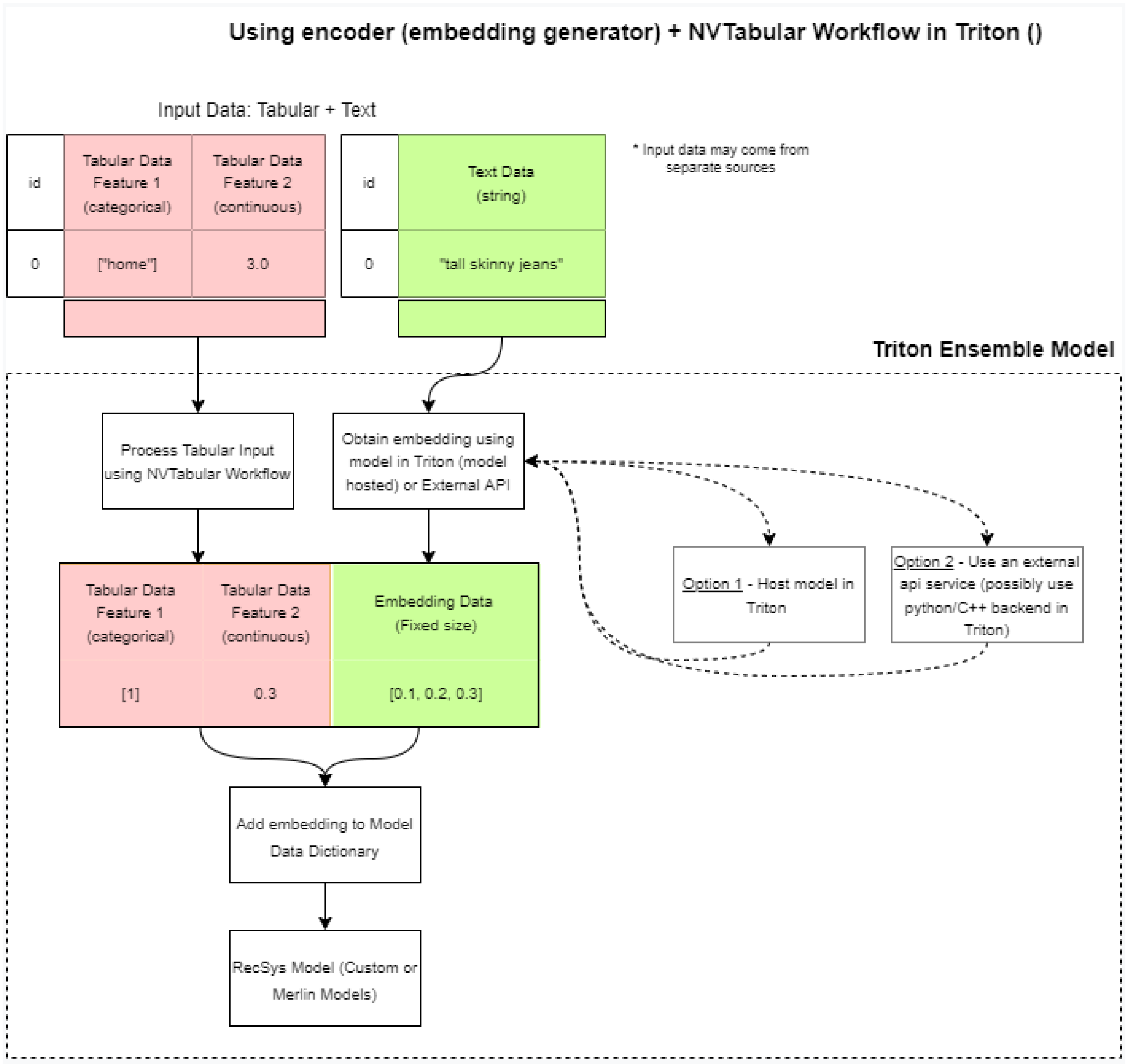

Using Model Embeddings in NVTabular Data Loader Workflow (Training)
Raw Data (parquet format): Tabular + Text
Workflow does not need to generate the embeddings within NVTabular.
NVTabular does not have to deal with the text string itself, only the embedding vector
Standard NVTabular Pre-processing Workflow
Separate embedding module: Triton or External API
Option 1 - Host model in Triton
Option 2 - Use an external api service (possibly use python backend in Triton)
Option 3 - Process using custom workflow (customer driven)
Processed Tabular Data (parquet format) + NVTabular Workflow (save for inference)
Embeddings (stored as arrays in parquet format)
Join parquet files using NVTabular JoinExternal (do not save NVTabular Workflow)
Processed Data (parquet format): Tabular + Embeddings
Use KerasSequenceLoader with Embeddings Defined as Continuous Columns (Sequence loader handles all batching)
Send data to TF Model or Merlin Models for Training

id	Tabular Data Feature 1 (categorical)	Tabular Data Feature 2 (continuous)
0	["home"]	3.0
1	["home","bath"	4.2
2	["bath"]	5.5

id	Text Data (string)
0	"tall skinny jeans"
1	"skinny jeans size 30"
2	"bedroom night stand"

id	Tabular Data Feature 1 (categorical)	Tabular Data Feature 2 (continuous)	Embedding Data (fixed size)
0	[1]	0.3	[0.1, 0.2, 0.3]
1	[1, 2]	0.42	[0.15, 0.25, 0.35]
2	[2]	0.52	[0.17, 0.27 0.37]

Using encoder (embedding generator) + NVTabular Workflow in Triton ()


Using encoder (embedding generator) + NVTabular Workflow in Triton ()
Input Data: Tabular + Text
id
Tabular Data Feature 1 (categorical)
Tabular Data Feature 2 (continuous)
0
["home"]
3.0
id
Text Data (string)
0
"tall skinny jeans"
* Input data may come from separate sources
Triton Ensemble Model
Process Tabular Input using NVTabular Workflow
Obtain embedding using model in Triton (model hosted) or External API
Option 1 - Host model in Triton
Option 2 - Use an external api service (possibly use python/C++ backend in Triton)
Tabular Data Feature 1 (categorical)
Tabular Data Feature 2 (continuous)
Embedding Data (Fixed size)
[1]
0.3
[0.1, 0.2, 0.3]
Add embedding to Model Data Dictionary
RecSys Model (Custom or Merlin Models)