Class DataProcessor

Table of contents

- Defined in File data_processor.hpp

Class Documentation

class DataProcessor

Data Processor class that processes operations. Currently supports CPU based operations.

Public Functions

inline DataProcessor()

Default Constructor.

InferStatus initialize(const MultiMappings &process_operations, const std::string config_path)

Checks the validity of supported operations.

Parameters

- **process_operations** – Map where tensor name is the key, and operations to perform on the tensor as vector of strings. Each value in the vector of strings is the supported operation.
- **config_path** – Path to the processing configuration settings

Returns

InferStatus with appropriate code and message

InferStatus process_operation(const std::string &operation, const std::vector<int> &in_dims, const void *in_data, std::vector<int64_t> &processed_dims, DataMap &processed_data_map, const std::vector<std::string> &output_tensors, const std::vector<std::string> &custom_strings)

Executes an operation via function callback. (Currently CPU based)

Parameters

- **operation** – Operation to perform. Refer to user docs for a list of supported operations
- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer
- **processed_dims** – Dimension of the output tensor, is populated during the processing
- **processed_data_map** – Output data map, that will be populated
- **output_tensors** – Tensor names to be populated in the out_data_map
- **custom_strings** – Strings to display for custom print operations

Returns

InferStatus with appropriate code and message

InferStatus process_transform(const std::string &transform, const std::string &key, const std::map<std::string, void*> &indata, const std::map<std::string, std::vector<int>> &indim, DataMap &processed_data, DimType &processed_dims)

Executes a transform via function callback. (Currently CPU based)

Parameters

- **transform** – Data transform operation to perform.
- **key** – String identifier for the transform
- **indata** – Map with key as tensor name and value as data buffer
- **indims** – Map with key as tensor name and value as dimension of the input tensor
- **processed_data** – Output data map, that will be populated
- **processed_dims** – Dimension of the output tensor, is populated during the processing

Returns

InferStatus with appropriate code and message

InferStatus compute_max_per_channel_cpu(const std::vector<int> &in_dims, const void *in_data, std::vector<int64_t> &out_dims, DataMap &out_data_map, const std::vector<std::string> &output_tensors)

Computes max per channel in input data and scales it to [0, 1]. (CPU based)

Parameters

- **operation** – Operation to perform. Refer to user docs for a list of supported operations
- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer
- **out_dims** – Dimension of the output tensor
- **out_data_map** – Output data buffer map
- **output_tensors** – Output tensor names, used to populate out_data_map

InferStatus scale_intensity_cpu(const std::vector<int> &in_dims, const void *in_data, std::vector<int64_t> &out_dims, DataMap &out_data_map, const std::vector<std::string> &output_tensors)

Scales intensity using min-max values and histogram. (CPU based)

Parameters

- **operation** – Operation to perform. Refer to user docs for a list of supported operations
- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer
- **out_dims** – Dimension of the output tensor
- **out_data_map** – Output data buffer map
- **output_tensors** – Output tensor names, used to populate out_data_map

InferStatus print_results(const std::vector<int> &in_dims, const void *in_data)

Print data in the input buffer in float32. Ideally to be used by classification models.

Parameters

- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer

InferStatus print_results_int32(const std::vector<int> &in_dims, const void *in_data)

Print data in the input buffer in int32 form. Ideally to be used by classification models.

Parameters

- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer

InferStatus print_custom_binary_classification(const std::vector<int> &in_dims, const void *in_data, const std::vector<std::string> &custom_strings)

Print custom text for binary classification results in the input buffer.

Parameters

- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer
- **custom_strings** – Strings to display for custom print operations

InferStatus export_binary_classification_to_csv(const std::vector<int> &in_dims, const void *in_data, const std::vector<std::string> &custom_strings)

Export binary classification results in the input buffer to CSV file using Data Exporter API.

Parameters

- **in_dims** – Dimension of the input tensor
- **in_data** – Input data buffer
- **custom_strings** – The comma separated list of strings containing information for the output CSV file. It should include application name as a first string required for the Data Exporter API and column names.