UM11543

Machine learning component user manual

Rev. 1 — 27 January 2021 User manual

Document information

Information	Content
Keywords	Content Component Library, Machine Leaning, support vector machine, Gaussian Mixture model
Abstract	Getting started with Machine Learning (ML) software component

1 Prerequisites

Prior to using this platform agnostic component library, the machine learning component, this document assumes the user is familiar with the:

- Chosen microcontroller unit (MCU)
- Corresponding software development kit (SDK)
- Cross-compilation tool chain to integrate Machine Learning (ML) software components defined for sensors.

The software machine learning component example application provides enough information to integrate to the underlying SDK. The component library implementation focuses on sensor specific libraries, sources, and interfaces, not the underlying MCU drivers, which are normally available and used directly from the MCU SDK implementation.

2 Overview

The machine learning software component is a development model that provides a comprehensive interface and its implementation for machine learning models such as SVM and GMM. Each model provides training interfaces to train sensor feature data. In order to determine the optimal coefficient for each model for the given data point, this sensor feature data can be raw, statistical data (mean, median, standard deviation etc.) or sensor fusion data. The component also provides an interface to generalize the prediction function depending on application conditions to be monitored. The platform interface provides abstraction to underlying communication drivers in the SDK, tool chains, and MCUs.

2.1 ML software component design

The main focus of the ML component is on machine learning applications for edge devices. The design addresses how the various ML software components should be designed in order to be platform agnostic and modular, with emphasis on low-power consumption and memory foot print.

The component defines a uniform interface for all ML components that includes initialization, training, and prediction interfaces for implementation in each ML component. Initialization function indent initializes the component with handler as an argument to keep to the instance for the rest of the interfaces. The training function indent trains the appropriate model based on the training data passed, which can be raw, statistical or fusion sensor data. The training function indent returns the coefficient of the trained model. The prediction function is passed with a function pointer which an application developer can use to create a user-defined prediction algorithm based on the trained data.

The interfaces abstract the application layer. The implemented algorithm should be agnostic to microcontroller interfaces, SDK interfaces, OS interfaces, and compiler dependencies. The user application is responsible for handling the multi-threading synchronization and resource handling. The component is designed to work seamlessly in any SDK environment and with application resource handlers.

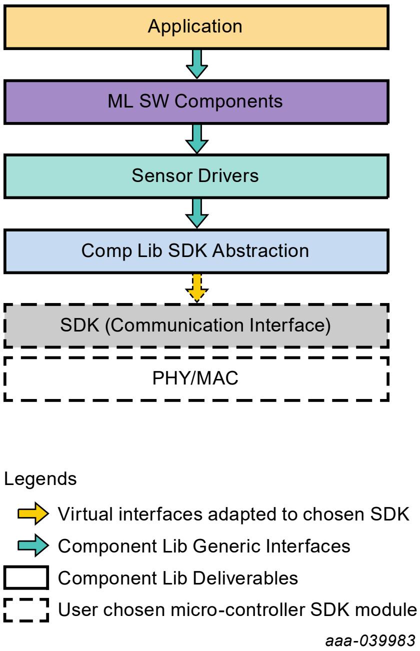

Figure 1. Machine language component design

3 ML component integration with product/user application

The ML components are designed to be microcontroller agnostic. This section describes details about ML component Integration strategy, the directory structure, and content overview. For details about ML interfaces, refer to API_Reference_Manual.

3.1 Integration

- Source files are designed to compile with platform agnostic product/user application software. Minor changes may need to be performed to support the compiler/IDE of the user/product application.
- Interfaces defined as xx_interface.h are the expected functions to be used in the product/user application software
- Components libraries are provided with the NXP MCUXpresso SDK integration example application. The integration test example applications demonstrate how to integrate platform agnostic component libraries with underlying microcontroller SDK communication interfaces using virtual interface abstraction provided by component libraries.

Note: *Before importing component library example projects for the standalone MCUXpresso IDE, the MCUXpresso IDE requires the corresponding microcontroller SDK package to be downloaded and installed on the IDE.*

3.2 ML component directory structure

This section provides a snapshot of the ML component directory structure.

```
Ml/
|--gmm /
    |-- src[1]
    |    |-- Source and Headers[2]
    |-- example[3]
    |    |-- MCUXpresso[4]
    |    |    └── <project_name>[5]
|--svm/
    |-- src[6]
    |    |-- Source and Headers[7]
    |-- example[8]
    |    |-- MCUXpresso[4]
    |    |    └── <project_name>[9]
|-- docs[10]
    |-- CompLib_ML_UG.docx[11]
    |-- ML_API_Reference_Manual.zip[12]
```

```
[1]Folder containing GMM component source files.
[2]Files containing SVM implementations.
[3]Folder containing GMM integration example with MCUXpresso SDK.
[4]Component libraries are provided with the NXP MCUXpresso SDK integration
 example application. The integration test example applications demonstrate
 how to integrate platform agnostic component libraries with underlying
 microcontroller SDK communication interfaces using virtual interface
 abstraction provided by component libraries.
[5]Folder containing GMM integration example for with MCUX.
[6]Folder containing SVM component source files.
[7]Files containing SVM implementations.
[8]Folder containing SVM integration example with MCUXpresso SDK.
[9]Folder containing SVM integration example for with MCUX.
[10]Folder containing release documentation for SVM component.
[11]ML Component user manual.
[12]ML Component API RM.
```

3.3 ML component testing

The ML component provides a platform independent implementation. The user is responsible for updating the the communication interface to visualize output using the SDK implementation for underlying microcontroller peripherals such as UART. NXP has limit tested the ML component for NXP microcontrollers FRDM-K64F (Cortex M4F core) integrated with MCUXpresso SDK. The reference example project for testing the ML component integration with MCUXpresso SDK is available under the “example” folder.

When the example is executed, the OCSVM component is first trained on training data that passes through the training function in order to produce the training model. The trained ML model learns the boundaries of normal cases and through learning. The model can predict anomalies in the test data by classifying test data outside the boundary as "outliers".

4 Revision history

Table 1. Revision history

Revision number	Date	Description
1	20210127	Initial release

5 Legal information

5.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

5.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

5.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Tables

Tab. 1. Revision history 5

Figures

Fig. 1. Machine language component design 3

Contents

1 Prerequisites 2
2 Overview 2
2.1 ML software component design 2
3 ML component integration with product/ user application 3
3.1 Integration 3
3.2 ML component directory structure 4
3.3 ML component testing 4
4 Revision history 5
5 Legal information 6

Date of release: 27 January 2021
Document identifier: UM11543