221e
Intelligent Precision Sensing

Expert-guided ML: Bringing AI to Your Embedded Microcontroller

Walter Zigliotto, Solution Specialist
Eran Belaish, CMO

Whitepaper

Expert-guided ML is the combination of subject-matter expertise with in-depth knowledge of Machine Learning techniques to achieve your goals.
How does Expert-guided ML work?

- Physical understanding of the data
- Signal pre-processing and feature extraction
- Machine Learning technique selection
- Embedding the algorithm into your device

Here you can find an example of Expert-guided ML software.

Introduction

Integrating Machine Learning (ML) algorithms is crucial for making your solutions more innovative. The introduction of ML to embedded Internet of Things (IoT) devices, known as tinyML, is fundamental for keeping up with current market demands. Nevertheless, implementing ML is a difficult task.

You might not have access to big data to train your ML algorithm. Your computational resources could be limited. You might be unable to achieve adequate accuracy. And sometimes, you don't even know where to begin implementing Artificial Intelligence (AI).

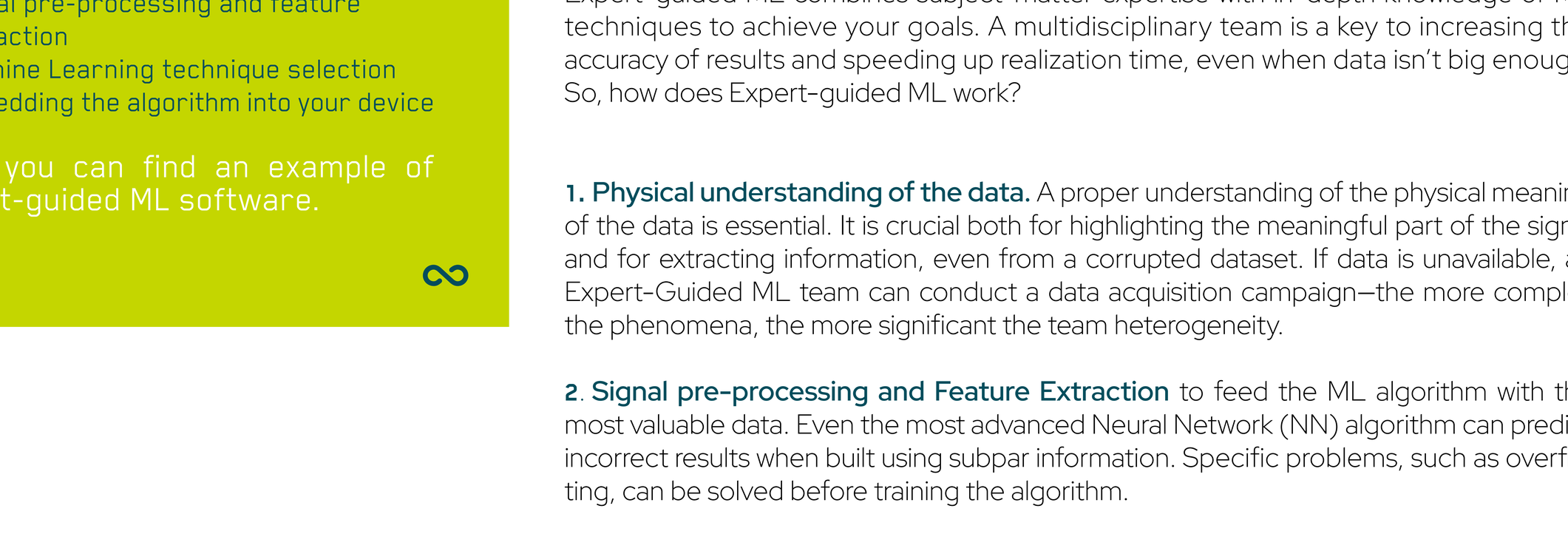
This is where Expert-guided ML comes in handy.
Expert-guided ML combines subject-matter expertise with in-depth knowledge of ML techniques to achieve your goals. A multidisciplinary team is a key to increasing the accuracy of results and speeding up realization time, even when data isn't big enough. So, how does Expert-guided ML work?

1. Physical understanding of the data. A proper understanding of the physical meaning of the data is essential. It is crucial both for highlighting the meaningful part of the signal and for extracting information, even from a corrupted dataset. If data is unavailable, an Expert-Guided ML team can conduct a data acquisition campaign—the more complex the phenomena, the more significant the team heterogeneity.

2. Signal pre-processing and Feature Extraction to feed the ML algorithm with the most valuable data. Even the most advanced Neural Network (NN) algorithm can predict incorrect results when built using subpar information. Specific problems, such as overfitting, can be solved before training the algorithm.

3. Find and implement the best ML technique for your requirements.

4. Integrate the resulting algorithm into your device while considering computational resource limitations.



Expert-guided ML: Bringing AI to Your Embedded Microcontroller

221e
Intelligent Precision Sensing

In the signal pre-processing and feature extraction stage, you will use the most powerful statistical tool available to filter and extrapolate the characteristics of the signal and the patterns you are interested in. During this step, you will also prepare the input for the ML algorithm.

1. Physical Understanding Of The Data

Data comes from multiple diverse sources. You may be interested in motion signals (e.g., angular velocity, GPS), vibration signals (e.g., acceleration), audio signals, biological signals (e.g., Photoplethysmography, Electrocardiogram) or optical signals, among others. Sometimes you may desire a combination of various signals depending on the monitored phenomena.

Regardless of the nature of the signals, properly understanding the raw data is essential to a successful project. For this reason, visualizing the phenomenon in the simplest way makes it easier to understand it thoroughly. Taking your time in this stage will allow to recognize characteristic patterns specific to the phenomena you are analysing. It will also save time in the following stages.

2. Signal Pre-Processing And Feature Extraction

This is the out-and-out data analysis part. In this stage, you will use the most powerful statistical tool available to filter and extrapolate the characteristics of the signal and the patterns of interest. Whether you use small or big data, try to increase its variability (e.g., splitting the dataset randomly and slicing it). Avoid potential problems like overfitting while attempting to reach the highest accuracy possible. The desired outcome is an algorithm that is as general as possible.

In this step, you will quantify the properties of the signals in the time and frequency domains. Once the frequency behaviour is known, you can filter the signal in the proper range and start extracting features. During this process, always take into account that in embedded applications, you might have limited resources in terms of computational power and time. For example, a real-time algorithm might not allow analysis with the statistical instruments of your choice. Nevertheless, once you select a valuable number of features, there are techniques (e.g., Principal Component Analysis (PCA)) that can guide you in selecting the most significant ones. The fewer features, the less complex your algorithm will be.

Finally, balance the dataset in the training set, the test set, and the validation set properly. There is no mechanical way to solve this issue; only experience and teamwork can help you to improve your performance.

Expert-guided ML: Bringing AI to Your Embedded Microcontroller

Selecting the best ML technique for your project requires balancing accuracy, computational time, memory space and power consumption.

A good balance ensures high accuracy without consuming too many resources that might be scarce in your embedded system.

3. Machine Learning Technique Selection

In the past few years, ML library availability has increased noticeably, along with computational power. Machine learning techniques are becoming crucial even in embedded applications where computational resources are limited. Choosing the proper method to adopt in an embedded IoT device means finding the optimal trade-off between accuracy, computational time, memory space and power consumption.

When designing an ML algorithm, try to reduce it to the bone to find the simplest and most efficient way to resolve the task. You can always increase the complexity of your architecture later. However, increasing complexity will not always end in improved performance. For example, the accuracy of a NN classifier does not always depend on its complexity (number of layers). There are other parameters you can set in a NN before increasing its number of layers. Sometimes if the results are not as expected, the problem might not be related to the complexity of the ML algorithm. It can derive from incorrect pre-processing and feature extraction of the training data.

4. Embed The Algorithm Into Your Device

If you have developed the algorithm offline, it will be necessary to port it into your device. This process is called “porting”.

If you were conscious of the limited resources of your embedded system from the onset, the process would typically consist of a simple translation between programming languages.

You can check the integrity of the embedded algorithm by running it with different test datasets. If the results are comparable with its offline version, you may test it directly on your board. If that is not the case, you may end up with the nightmare of being unable to embed the solution into your device. In this case, you might want to check our MakeSense program here.

Putting it all together, here you can find an example of Expert-guided ML software that uses the above methodologies.

Expert-guided ML: Bringing AI to Your Embedded Microcontroller

221e
Intelligent Precision Sensing

Summary

Expert-Guided ML is the result of combining expertise in different fields with in-depth knowledge of Machine Learning techniques to achieve a wide variety of goals. It generally works according to these steps: physically understanding the data, signal pre-processing and feature extraction, ML technique selection, and algorithm embedding.

The first stage is fundamental to recognizing characteristic patterns specific to the analysed phenomena. It is also a significant time saver. In the signal pre-processing and feature extraction stage, utilize the most powerful statistical tool to filter and extrapolate signal characteristics and patterns of interest. This step also prepares the input data for the ML algorithm. The third phase is choosing the right ML architecture to optimize the performance and accuracy of the results. Finally, you will embed the algorithm into your device. Although last, it is crucial to keep it in mind throughout the process to embed the finalized algorithm successfully.

To learn more about sensor fusion hardware, software and AI solutions, please visit www.221e.com and follow us on LinkedIn here.

If you need assistance with sensing or AI, please visit our MakeSense program here.

To learn more about sensor fusion hardware, software and AI solutions, please visit www.221e.com and follow us on LinkedIn here.