Streamlining the development and implementation of Deep Learning Machine Vision applications with end-to-end data centric tool

Today, Deep Learning algorithms have become widespread and relatively easy to run. Moreover, they have shown their potential as a new machine vision tool to reach new quality assurance and/or efficiency objectives for various types of industries and processes (e.g. pharma, food, raw materials, semiconductor industry, …). In some cases, the objectives reached by Deep Learning algorithms were hard or not at all achievable with traditional machine vision approaches.

However, developing new applications based on Deep Learning can still represent a significant challenge. For instance, just using the latest state-of-the-art Deep Learning model or algorithm will not magically solve your problem. To maximize your chance to develop a successful application based on Deep Learning algorithms, we strongly recommend following a **data centric approach**.

The data centric approach means that you must foremost tackle your problem by gathering and working with your data and have systems to properly manage and track it. In a Deep Learning application, the data is not limited to the input images that you want to process. It is also the annotations, i.e., the data you want to predict from your images. It is the result of your training sessions that you will use to compare and select the best model. It is all the metadata characterizing your images (ROI, masks, etc.), how they will be used (dataset splits) and/or how the training will be performed (data augmentation settings, model settings, etc.).

This is one of the main directions that the machine vision software industry has taken to help their customers overcome the challenges of Deep Learning applications. For instance, let's have a look at how the **Euresys Deep Learning Studio** application helps in that approach. Deep Learning Studio is the graphical tool for managing and annotating data and training Deep Learning models that can be used with the Euresys Open eVision libraries. It is a **free desktop application**. This means that you can directly work and see what you can do with your data without having to pay anything and without having to send your confidential images across the internet to use an online service.

When tackling a new Deep Learning application, the first step will be to select what you want to predict from your images. Various criteria can apply such as the size of your images, the number of elements that you want to detect in your images, whether the elements can overlap or not , the level of precision that you need, etc. For example, in Deep Learning Studio, you have 5 types of Deep Learning tools that you can choose from: EasyClassify for image classification, EasySegment Unsupervised for anomaly detection trained only on good images, EasySegment Supervised for precise pixel-level segmentation, EasyLocate Bounding Box for the localization and identification of elements of different sizes and EasyLocate Interest Point for the localization and identification of elements having the same size.

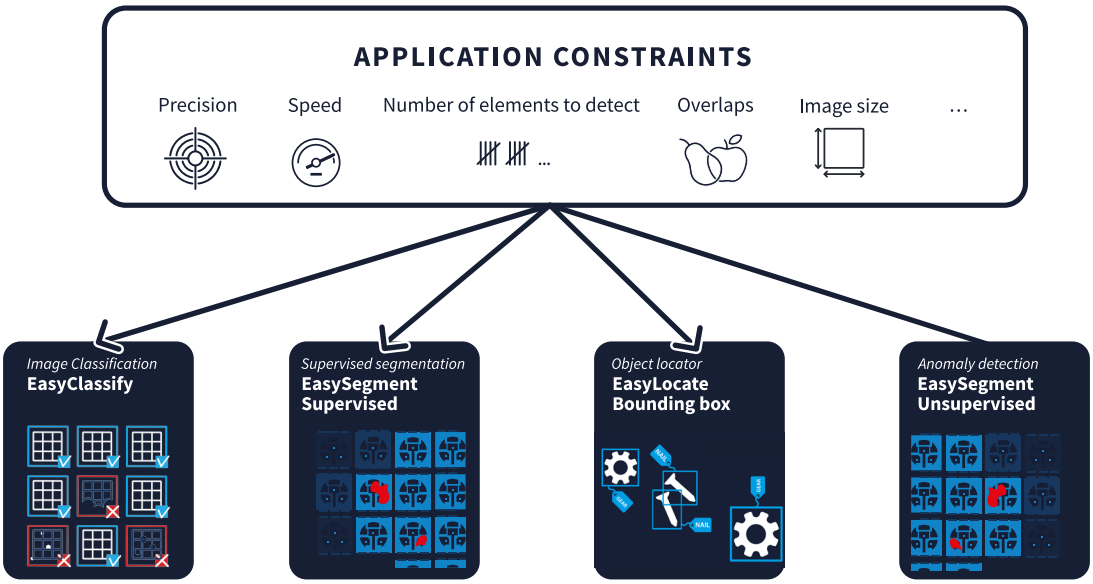

FIGURE 1 *Criteria influencing the choice of a Deep Learning tool*

Selecting the most appropriate tool for your data is essential. Let's take the inspection of contaminants in coffee beans. The segmenter tool, EasySegment, was here retained for it allows not only to detect the presence or absence of contaminants but also provides the location and surface area of it/them within the image as they need to be removed by another process to prevent contamination and/or process machinery excessive wear.

FIGURE 2 *Inspection of contaminants in coffee beans*

The next step is to gather and annotate your images. A tool such as Deep Learning Studio will help track the images you already have in your dataset and which images are annotated. We strongly recommend prioritizing consistency when annotating your images. For instance, for the coffee dataset, we could consider a foreign object as a defect only when its size is large enough to cause a problem on the production line. In this case, we should still annotate all foreign objects regardless of their size to ensure that the training data is consistent. The decision to reject or accept the foreign object is then made by analyzing the size of the foreign objects in the results.

FIGURE 3 *Annotation consistency is very important.*

In Deep Learning Studio, many advanced machine learning best practices and concepts are integrated transparently or made part of the application workflow. For example, data balancing is about correctly training a model when the number of images we have for each label is very different. This is especially important in defect detection applications where the number of good samples can be several orders of magnitude larger than the number of defective samples. Data balancing is integrated at the core of Deep Learning Studio through a single parameter per label called the "label weight". All the intricacies of the data balancing techniques are thus handled automatically. This kind of integration makes training smoother and less prone to error.

Another example would be dataset splitting. It is a good practice in machine learning to ensure that the result of a training is robust and can be applied to new, not yet seen, images. It consists in separating the images in three subsets: the "training set", the "validation set", and the "test set". The "training set" contains the images that will be used to directly train the neural network. The "validation set" contains the images that will be used to select the best neural network during training. The "test set" contains images that will never be used during training, but will be used afterwards to make sure that the trained model performs well on new images. Generally, the "training set" will contain more images than the "validation set", which itself contains more images than the "test set". However, in machine vision applications where getting images can be complex, every image counts and different splits can thus have an influence on the result. For instance, if a rare defect is only represented by a few images in the dataset and all these images are put in the test set, the model is not able to learn anything about this defect. It is thus important to have tools to create, track, and manage these dataset splits. In Deep Learning Studio, this is integrated in the workflow of the application at the same level as the annotation of the dataset or the training of a model.

A program capable of managing your images, training your models and analyzing the results allows for advanced and **iterative model and dataset building**. Indeed, complex problems can require a lot of images to make a Deep Learning model work. In part because you don't know beforehand which images will represent a tough problem for the neural network. For instance, with the coffee inspection dataset available along Deep Learning Studio: by first annotating only 10% to 20% of the images and training a model on these, we can obtain an EasySegment Supervised model that will already output correct results for 30% of the remaining images, and mostly correct results for another 30%. In the end, only 40% to 50% of the images were completely annotated manually. All the other annotations were directly produced by intermediate Deep Learning models or adapted from "good enough" results.

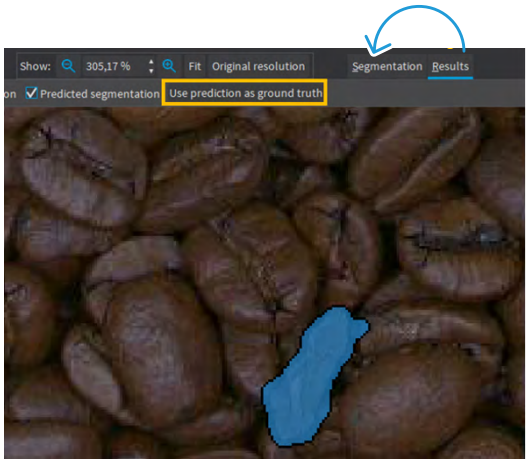

FIGURE 4 *In Deep Learning Studio, you can import a correctly predicted result as ground truth with a click of a button*

Of course, the challenges in using Deep Learning techniques to solve machine vision applications are not limited to data management. Designing or choosing a neural network architecture to reach the required performance or processing time can be complex. A machine vision library such as Euresys Open eVision proposes custom designed neural networks optimized for the typical problems encountered in the machine vision industry.

Deployment of the Deep Learning model in production can also be challenging:

- How to integrate the Deep Learning model in your program?
- How to ensure that it will work consistently on different platforms *(Windows/Linux, x86/ARM/GPU)*?
- How to combine the Deep Learning models with other machine vision operations?

This is where a full-featured industrial machine vision library like Euresys Open eVision excels. The API available for several programming languages allows you to integrate Deep Learning algorithms along with code readers or traditional pattern finders in your application. It simplifies, for instance, the maintenance and evolution of your application by having fewer dependencies. For example, a Deep Learning model for the inspection of foreign material in coffee beans was deployed in fewer than 10 lines of code on a NVIDIA Jetson Xavier NX based smart camera running at 10 fps for 512x512 images.

In conclusion, we have seen that data is the core of any Deep Learning based application and that the tools for managing your data and working with the Deep Learning algorithm are paramount to get good results. The Euresys Open eVision library and Deep Learning Studio program are a powerful implementation of this methodology.

Antoine Lejeune
Software Engineer - Euresys

MORE INFORMATION

https://www2.euresys.com/Deep-Learning-Bundle

OR TUTORIAL VIDEOS

Deep Learning Training Part 1/3

Introduction and basic usage of Deep Learning Studio

Deep Learning Training Part 2/3

How to get the most out of your data?

Deep Learning Training Part 3/3

How to use and get the most out of each DeepLearning tool?