Deep Learning in Automotive Software

Journal:	*IEEE Software*
Manuscript ID	SWSI-2016-10-0152.R1
Manuscript Type:	SWSI: Automotive Software
Date Submitted by the Author:	21-Dec-2016
Complete List of Authors:	Lami, Giuseppe; Istituto di scienza e tecnologie dell'informazione Alessandro Faedo, System and Software Evaluation Center Falcini, Fabio; Istituto di scienza e tecnologie dell'informazione Alessandro Faedo, System & Software Evaluation Center Mitidieri, Alessandra; Fiat Chrysler Automotive, EMEA PRODUCT DEVELOPMENT
Keywords:	D.2.18.a Life cycle < D.2.18 Software Engineering Process < D.2 Software Engineering < D Software/Software Engineering, I.2.10 Vision and Scene Understanding < I.2 Artificial Intelligence < I Computing Methodologies, D.2.0.d Standards < D.2.0 General < D.2 Software Engineering < D Software/Software Engineering, D.2 Software Engineering < D Software/Software Engineering
Abstract:	In automotive software, the deep learning-based systems, with their peculiar features, are playing an increasingly pervasive role. As a result, inside the automotive software engineering community, an awareness of the need of integrating the deep learning development approach with the "traditional" ones is growing, at technical, methodological and cultural levels. In particular, the innovative and data-intensive phase of deep neural network training, by means of ad-hoc training data, is pivotal in the software development of vehicle functions that rely on such a technology paradigm and thus necessitates a special focus. A development lifecycle fitting deep learning needs is introduced and an improvement initiative, based on Automotive SPICE, for an effective adoption of DNN in the automotive software applications is presented.

Deep Learning in Automotive Software

As present-day automobiles are becoming so computerized, it is not surprising that Artificial Intelligence (AI) is entering into some of the main on-board Electronic Control Units (ECU). A key factor is the growing presence in latest car models of sophisticated sensor technology that is accelerating the evolution of the so-called automotive Advanced Driver Assistance Systems (ADAS) [1].

ADAS evolution, with all its six progressive levels of driving automation support (ranging from 'no automation' to 'full automation' according to Society of Automobile Engineers), is so paving the path towards the general availability of well-advertised applications that include pedestrian detection, adaptive front lighting and, of course, autonomous driving in mid and long-term perspectives. This is why a highly complex technology such as machine learning is needed for a computer to assist or replace a human driver in a car.

Deep Learning (DL) is now considered by many automotive companies as a mature and viable technology, not only in the improvement of ADAS performances, but also in making progresses in other functional domains such as, for example, engine management [2] and vehicle cyber-security protection [3]. Deep learning technology, therefore, is not going to be confined only to the ADAS functions.

Deep learning is a branch of machine learning based on a set of Artificial Neural Network (ANN) algorithms that model high-level abstractions in input data by using a deep graph representation made of multiple processing layers. In computer vision-based ADAS, for example, deep learning algorithms improve the detection and recognition of multiple objects, support object classification and enable recognition as well as prediction of actions.

At the same time, the introduction of deep learning technology on-board cars is starting to have a significant effect on the automotive software engineering that needs to incorporate and harmonize the development of these technologies in its current state-of-the-art. In fact, it is important to remind that, since much more than a decade, automotive Original Equipment Manufacturers (OEM) have been requiring the automotive software to be developed in a well-engineered, regulated and controlled way.

Granted that deep learning is a well-articulated and already developed discipline at both academic and application levels, it is fundamental to understand whether the state of the art of deep learning is aligned with automotive demands, at both technical and methodological levels. A first glance answer prompts: "Not entirely for sure".

In fact, while machine learning is successfully pushing the development of new hardware components and new hardware architectures, the automotive software engineering is at the forefront of this innovation. This new challenge finds its peak in the data-intensive training process that is required by the deep learning-based systems. The idea of solving a problem through the training of an artificially intelligent system, instead of targeting the solution using domain knowledge (i.e. feature engineering), is revolutionary for automotive software applications.

From the field

Unsurprisingly the penetration of Electronic Control Unit (ECU) hosting artificial intelligence is supposed to grow in a steady and substantial way according to endorsed market researches [4].

The expected volumes are so important to reinforce the need to analyze the peculiarities and to integrate deep learning in the development lifecycle of automotive electronics systems.

Google is making remarkable and highly visible investments in the development of autonomous vehicles. Its prototypal self-driving vehicles embed deep learning based technology that is already able to detect pedestrians in various and challenging scenarios. Google deep learning systems have achieved outstanding performance, making the error rate for machine vision lower than the one of a human being (5% error rate is human benchmark). This achievement, also due to new hardware architectures using multiple Graphics Processing Units (GPU), is pushing the migration of features based on traditional image processing technology to deep learning-based solutions.

This is just the beginning and, even so, remarkable elements of artificial intelligence are already available in circulating vehicles. In the ADAS domain, Tesla is reported to feature onboard the implementation of a neural network functionality for vision, sonar and radar processing that runs on the powerful NVIDIA DPX2 processor in the driving control unit [5].

Several other suppliers are already active players. The Intelligent Surround View (ISV) system, by AdasWorks, is an example of an ADAS neural network-based implementation, which processes the environment around the car using the visual information coming from several cameras [6]. In addition, DENSO R&D labs together with other important companies R&D labs are actively researching in this direction.

In the infotainment domain, the 2015 BMW 7 Series is reported to be the first car to feature an innovative voice recognition solution based on deep learning technology that works also in absence of the wireless car connectivity [5].

Other automotive applications of deep learning, currently under development, range from engine fault diagnosis and emissions management to detection of vehicle network intrusion.

From an hardware perspective, the level of electronic support required to embed deep learning in high-performance and safety-related automotive applications is so demanding that companies are aggressively developing new generations of chips: an example is the Mobileye's upcoming cutting-edge EyeQ5 proprietary chip. In regard of commercially available electronics components, Intel is positioning with the new Xeon Phi chip to compete in this market, which has been so far ruled by Nvidia Tegra chip.

Deep Learning

The most essential property of artificial neural networks is the ability to improve their problem-solving capabilities through a "learning" process triggered by exemplary input [7]. This feature is very helpful in scenarios in which there is no detailed, complete or predictable information about the problem as usually in automotive driving situations. Another relevant feature is their parallel

structure, which benefits from the use of powerful hardware to obtain timely and thus usable computation results.

Deep learning, which is synonymous of deep neural network (DNN), is a specific kind of artificial neural network able to model complex non-linear relationships using multiple hidden layers of units between the input and output layer (Figure 1 – right). Deep learning excels on finding patterns when input are massive analog data – this means not a few numbers in a tabular format, but instead images of pixel data or audio data. Until 2006 advances, DNNs were outperformed by shallow neural network that relied on feature engineering which is the embedding of the domain knowledge in the solution design.

The structure of a deep neural network is flexible and can be customized by selecting attributes such as the number of hidden layers, number of units per layer, number of connections per unit etc. These attributes, known as hyper-parameters, define the structure as well as the behavior of a deep learning-based system.

Summarizing, the fundamental characteristics of deep learning are:

- Input-output mapping through the learning process
- Nonlinearity - DNNs are composed of an interconnection of nonlinear computational elements (a.k.a. neurons or nodes)
- Adaptation capability - DNNs have a built-in adaptation ability to the changes in the environment
- Fault tolerance - due to DNN distributed nature, localized faults in hidden layers leads to a degradation of performance rather than a system failure.

Their processing capability is stored in the inter-unit connection weight obtained by a process of adaptation to a set of training patterns.

Convolutional neural networks (CNN), also known as ConvNets, are a type of deep neural network (Figure 1 – bottom) conceived to manage data in form of arrays with some degree of spatial structure.

They are designed to emulate the behavior of a visual cortex and perform very well on visual recognition tasks because the convolution operation (in shape of matrix products) itself is capable of capturing the features of images. Convolutional neural networks have special layers called convolutional layers and sampling layers that allow the encoding of the images properties. Essentially, a CNN transforms 3D input (e.g. an image with W rows, W columns, and 3 color channels) in a feedforward mode [7] along the network.

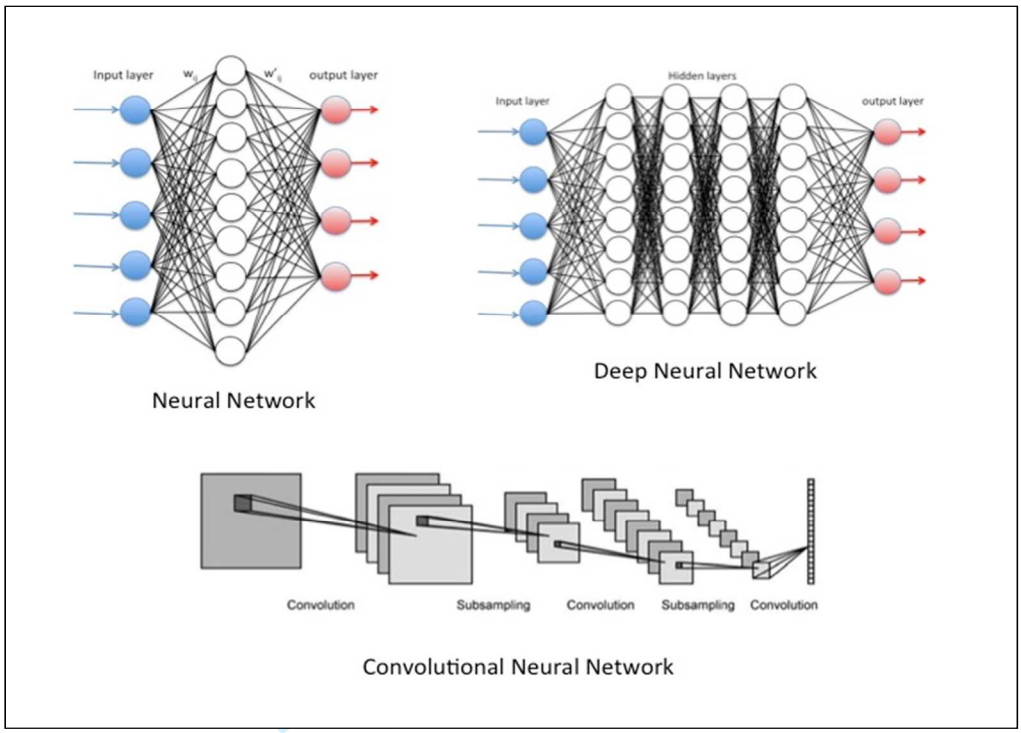

Figure 1: Examples of neural networks

Convolutional Neural Networks, for their characteristics such as input data segmentation and a high degree of parametrization (up to hundreds of thousands), are of special interest for visual applications in the automotive context such as object detection, vehicle detection, road marking detection and more.

Recurrent Neural Networks (RNN) is another relevant class of deep neural network where connections between units form a directed cycle [8]. They have been successfully used in speech recognition and natural language processing.

Moving from the theory to the software practice for a better understanding, the following source code fragment illustrates an instructive C implementation of some of the main deep learning concepts. In particular, trivial samples of the neuron and connection data structures and of a "create" function for a neuron are presented in Figure 2:

```
typedef struct _connection {
      float inter_connection_weight;
      struct _neuron * from;
} NN_connection;

typedef struct _neuron {
      float param;
      NN_connection * neuronConnections;
}NN_neuron;

NN_neuron* create_neuron(int n) {
      NN_neuron* newNeuron = malloc(sizeof(NN_neuron));
      newNeuron ->param = 0;

      connection * a = malloc(n*sizeof(NN_connection));
      newNeuron -> neuronConnections = a;
      return newNeuron;
}
```

Figure 2: Exemplary DNN source code portion

Actual automotive applications can easily feature DNNs with up to ten layers and thousands of nodes and their development benefit from the availability of consolidated software frameworks such as Theano, Caffe, Torch, Neon, TensorFlow, Deeplearning4J, CNTK, and many others, including the proprietary ones.

Training a Deep Neural Network

Keeping for simplicity computer vision as a reference, training deep multi-layered neural networks requires a large set of image frames (typically extracted from video clips of driving scenarios) featuring shapes, edges and colors in order to be able to match the input to what statistically is expected to be the correct result. This training process is a key element because it allows to fully exploit the deep neural network capability of detecting an object, taking into account also the context of the image.

The training of deep learning algorithms typically takes days to weeks to train, forcing projects to make compromises between accuracy and time to deployment. Three main training strategies are adopted [9]:

1. Supervised training - it relies on a sample of the environment knowledge based on training data in shape of pairs of input and their corresponding target output by means of appropriate labels
2. Unsupervised training - this approach leverages the statistical regularities present in the input data
3. Reinforcement training - it relates to taking actions in an environment to maximize a long-term reward, like a sort of trial and error approach.

Supervised training is sometimes effectively associated to reinforcement training in automotive applications.

In general, the identification of the suitable training method largely depends on the specific type of deep neural network and on the characteristics of the problem under consideration. According to authors' field experience, supervised training is widely used in DNN-based automotive developments.

The application of supervised learning in automotive requires massive annotated training datasets whose dimensions (i.e. the number of images) are not publicly available but their magnitude order ranges from hundreds of thousands to millions. The training dataset typically contains annotations or labels of positive and negative regions. Additional preprocessing can be applied to improve both the accuracy and the robustness of the training. A bounding box corresponding to pedestrians that precisely delimits the human body shape is a typical example of training data preprocessing (Figure 3).

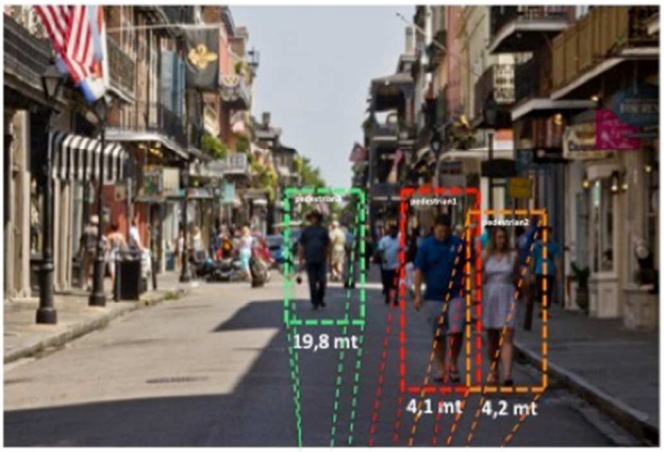

Figure 3: Annotated training data

Preprocessing has also weighty organizational impacts in terms of know-how and human resources to be dedicated to this time-consuming task.

While the training data set serves to train the learning algorithms, the validation data set is used to avoid over-fitting and thus to select the most suitable algorithm by comparing the different performances and decide which one to take. Over-fitting occurs when a model is excessively complex and begins to "memorize" training data rather than "learning" to generalize from trend.

The test data set is used to achieve the desired performance characteristics such as accuracy. These sets represent a valuable proprietary asset of suppliers and OEMs.

During the training phase, the associated learning algorithm populates an information structure, named classifier, by "learning from" the applied training data set. Accordingly, the knowledge of the environment available in the training set is transferred to the classifier and implicitly to the deep neural network itself.

In supervised training, several learning algorithms are available and the most popular is back-propagation [10]. According to the back propagation approach, the learning process has two phases. During the first one, a training input pattern is fed to the network input layer. The network then propagates the input pattern from layer to layer until the output pattern is generated by the output layer. If this pattern is diverse from the expected output, an error is computed and then propagated backwards through the network from the output layer to the input layer. The inter-unit connection weights of the network are modified as error is propagated.

It is thus unmistakable that the deep learning introduces major novelties compared to the consolidated automotive software engineering practices. The training itself actually functions as a programming activity.

A lifecycle for Deep Learning: the W model

The software side of DNN development is a highly iterative activity composed by a stream of steps:

1. DNN Requirements identification
2. Learning Algorithm Development
3. DNN Training
4. DNN Training Validation
5. DNN Validation

Compared with the traditional approaches, deep learning development process needs the support of empirical design choices driven by heuristics. Development often start from well-known learning algorithms, which have been proven effective in comparable problems or domains, since the understanding of the result of learning process is difficult to be grasped and thus managed.

Expected DNN requirements also include performance demands, expressed in terms of statistical benchmarking of the DNN functional behavior (i.e. error rate), that are carefully targeted during the DNN validation phase.

Development and adaptation of the appropriate algorithms, typically based on convolutional approaches, which enact the learning capability of the deep neural networks for automotive applications, are broadening the skill spectrum of various R&D labs. In fact, a notable presence of scientists, even at industrial level, that cooperates with engineers during the witnesses the specifics of this promising setting.

During the training, the learning algorithm is evaluated and the inter-unit connections weights are repetitively and experimentally adjusted; such a loop goes on until prediction accuracy does reach its target.

Automotive software engineering, while welcomes innovation and outstanding functional performances, is strict in its request for a robust and predictable development cycle. For that reason, it is important place deep learning in more controlled V-model perspective to address a lengthy list of challenges, such as requirements criteria for training, validation and test data sets, criteria for the training data pre-processing, management of very large sets of parameters and much more.

The introduction of a more structured conception of the deep learning life-cycle is instrumental to reach a controlled development approach that cannot be addressed by the mere functional benchmarking obtained with validation activities. It is essential to pursue both fundamental directions of the software engineering: high performance at functional level and high-quality development process.

However, deep learning intrinsically introduces its specific features (not completely fitting with the V-model for software development. The central role that is played by data in this context (e.g. for DNN training and training validation) makes essential the introduction of the W model and, to support it, we introduce the term “programming by example” to highlight the importance of data in developing systems based on deep learning technology.

The deep learning W-model is a framework lifecycle that conceptually integrates a V model for data development in the standard V perspective (Figure 4).

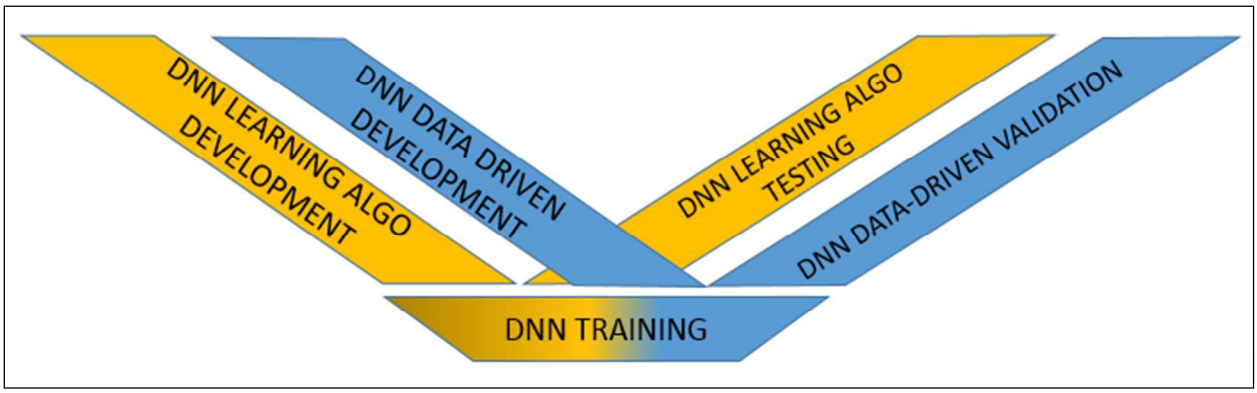

Figure 4: The W-model for deep learning

This lifecycle model acknowledges that deep learning is driven by software development as well as data development. The design and the creation of training/validation/test data sets, together with their exploitation, are crucial development phases because the DNN's functional behavior is the combined result of its architectural structure and of its automatic adaptation through a training process.

Deep learning moves away from feature engineering by definition and this element makes the W-model an appropriate and useful representation of this sophisticated paradigm.

Automotive Software Standards and Deep Learning

The software development process for on-board automotive ECUs is subject to proprietary OEMs norms as well as several international standards. Among them, the most relevant and influential standards for deep learning are Automotive SPICE [12] and ISO 26262 [13]. Needless to say that these standards are still far from addressing it with dedicated statements.

The Automotive SPICE standard - SPICE stands for Software Process Improvement and Capability dEtermination - provides a process framework that disciplines, at high level of abstraction, the software development activities and allows their capability assessment in matching pre-defined sets of numerous process requirements. ISO 26262, titled "Road vehicles – Functional safety" and released in late 2011, targets safety-related development and its scope expectedly includes system, hardware and software engineering. It is important to remark that the ISO 26262 standard already addresses configuration and calibration data, even though this aspect is an order of magnitude simpler and plainer than the development of DNN data sets.

Both standards, as far as the software is concerned, rely conceptually on the traditional development lifecycle: the V-model. It is also very relevant for deep learning the ISO PAS "Safety of the Intended Functionality (SotIF)" [14] that is currently in advanced development stage. This ISO document addresses the fact that for some ADAS applications there can be safety violations with a system free from faults - for example a false-positive detection by a radar of an obstacle for the vehicle – because it is extremely problematic to develop systems able to address every possible scenarios.

Because of its pervasive adoption and of its holistic coverage of the automotive software development processes, Automotive SPICE standard can be recognized as the appropriate reference for analyzing systematically the peculiarities of deep learning from the automotive software engineering point of view and for pushing a harmonized methodological maturation of deep learning practices.

The following picture, by placing the software part of the V-model of Automotive SPICE 3.0 on top of the coined W-model, hints at the need of initiating an organized evolution of deep learning along the process requirements of the Automotive SPICE (Figure 5). Same approach may apply for ISO 26262.

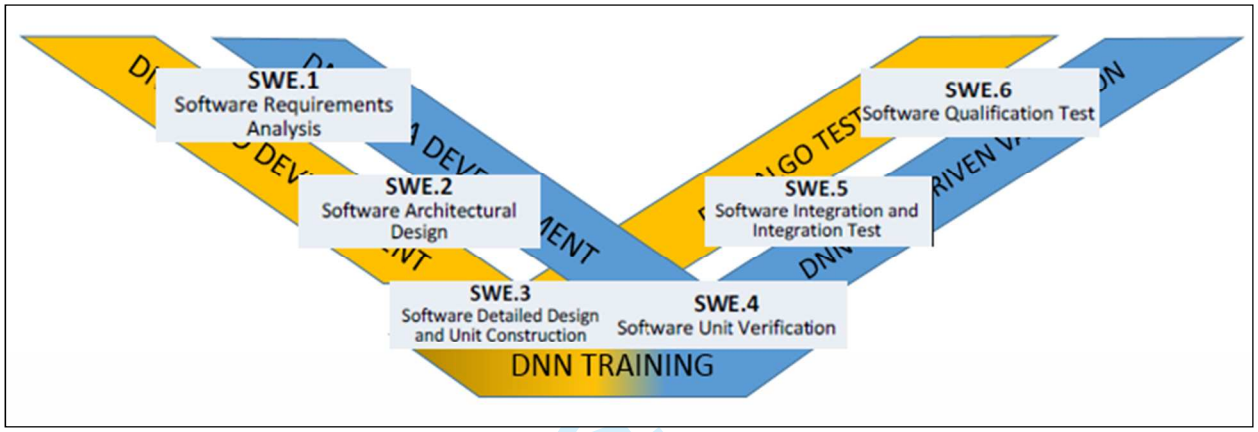

Figure 5: The W-model in Automotive SPICE 3.0 software perspective

Given the stringent requirements of the automotive market for the ECU development in terms of rigor and control, a substantial growth and a stabilization phase within deep learning is more than necessary. Just scratching the surface, important and unanswered methodological issues emerge, for instance the traceability infrastructure among the deep learning development stages.

This improvement path is thus pivotal to the general availability of the ambitious future achievements for automotive software such as fully automated driving.

The Way Ahead

As deep learning is ushering in a radical change in the automotive software field and the presented W-model for deep learning is a promising and understandable base for developing a beneficial and comprehensive integration of deep learning with the traditional automotive software engineering concepts.

The Italian automotive Software Process Improvement Network [15] has launched an open working group in order to facilitate the harmonization of deep learning practices in the automotive software engineering using the standards such as Automotive SPICE 3.0, the next edition of ISO 26262 and of the SotIF. Furthermore, the characterization of the introduced W-model is going to be addressed within this initiative.

References:

1. Jesse Levinson, ed alt. Towards fully autonomous driving: Systems and algorithms. Intelligent Vehicles Symposium (IV). IEEE 201.
2. A. Parlak, et alt. Application of artificial neural network to predict specific fuel consumption and exhaust temperature for a Diesel engine. Applied thermal Engineering. Vol. 26, Issues 8-9, pagg. 824-828. Elsevier 2006.
3. Min-Joo Kang, Je-Won Kang. Intrusion Detection System Using Deep Neural Network for In-Vehicle Network Security. 2016.
4. IHS Automotive Electronics Roadmap Report, H1 2016
5. Report at https://cleantechnica.com/2016/10/25/inside-nvidias-new-self-driving-supercomputer-powering-teslas-autopilot/
6. Report at http://ceva-dsp.mediaroom.com/2016-09-15-CEVA-and-AdasWorks-to-Demonstrate-Free-Space-Detection-for-Autonomous-Driving-at-AutoSens-Conference-2016
7. J. Schmidhuber. Deep learning in neural networks: An overview. Neural Networks Journal. Vol 61, pag. 85-117. Elsevier 2015.
8. S. Haykin, Neural Networks and Learning Machines. Prentice-Hall, 2009.
9. Jacopo Credi. Traffic sign classification with deep convolutional neural networks. Master's thesis in Complex Adaptive Systems. University of Tecnology of Gothenburg, 2016.
10. Yann LeCun, Yoshua Bengio & Geoffrey Hinton. Deep learning. Nature 521, 436–444, 28 May 2015.
11. Hui Miao, Ang Li, Larry S. Davis, Amol Deshpande. Lifecycle Management for Deep Learning. University of Maryland, 2015.
12. Automotive SPICE, Process Assessment Model v3.0, 2015.
13. ISO 26262 - Road Vehicles - Functional Safety, International Organization for Standardization, 2011
14. ISO/AWI PAS 21448 Road Vehicles -- Safety of the intended functionality
15. www.automotive-spin.it