Resource efficient machine learning for driver safety

Public report

Picture from project

Project within FFI Sub-Program “Electronics, Software and Communications”
Author Karl-Filip Faxén
Date 2023-01-30

Content

FFI in short

FFI is a partnership between the Swedish government and automotive industry for joint funding of research, innovation and development concentrating on Climate & Environment and Safety. FFI has R&D activities worth approx. €100 million per year, of which about €40 is governmental funding.

For more information: www.vinnova.se/ffi

Instructions

- *Instructions are written in brown Italic text and shall be removed.*
- *No limitation on number of pages.*
- *Do not insert documents or objects that must be opened before reading. It will not work since the report is turned into a PDF-file when uploaded.*
- *This report is public and will be published on FFI´s webpage.*

1. Summary

Around 90% of road traffic accidents are due to driver errors, and the driver was distracted in two thirds of all accidents. Hence EU regulations are expected to make driver monitoring systems mandatory for new vehicles by 2025. Driver monitoring is one of several sophisticated driver support systems enabled by the advances in machine learning. While machine learning is a powerful approach, its high computational resource requirements translate into increased component cost and energy consumption.

A machine learning model is parameterized by a large number of weights, up to several millions in popular deep neural networks. Pruning is a set of methods for training networks where a large fraction of these weights are zero. In principle, operations on these weights can be eliminated, saving computational resources. Such algorithms are known as sparse algorithms. In practice, sparse computation is nontrivial and conflicts with exploiting features of modern CPUs such as SIMD vector units and caches since skipping zeroes makes the computation less regular.

We have investigated sparse implementations of an important class of deep learning operators called convolutions that consume some 90-95% of the compute resources in popular image processing networks. In particular, we have focused on mapping sparse convolutions to efficient code for the ARM architecture that is often used to run driver assistance systems in current vehicles. Our code efficiently exploits caches and vector units, and typically outperforms well tuned dense implementations when the fraction of zero weights is above 40-50%.

2. Sammanfattning på svenska

Omkring 90% av alla trafikolyckor orsakas av förarmisstag, och i två tredjedelar av fallen var föraren distraherad. Därför väntas EU-regler kräva förarövervakningssystem för nya fordon från 2025. Förarövervakning är ett av flera sofistikerade stödsystem som möjliggjorts av framstegen för maskininlärning. Maskininlärning är en kraftfull samling metoder, men dess stora beräkningskrav driver ökade komponenkostnader och ökad energiförbrukning.

En maskininlärningsmodell är parametriserad av ett stort antal vikter, upp till flera miljoner för de populära bildbehandlingsnät vi har studerat. Beskäring (prining) är en samling metoder för att träna nät där många vikter är noll. Operationer på vikter som är noll kan ofta undvikas; multiplikation med noll ger alltid noll och en addition med noll behöver inte utföras. För att dra nytta av dessa möjligheter behövs glesa algoritmer och datastrukturer. Dessa minskar antalet operatoner som behöver utföras, men till priset av att de som blir kvar ofta är mer oregelbundna. Det ställer till problem för moderna processorer som är beroende av cacheminnen och (korta) vektoroperationer för att uppnå full prestanda.

Vi har studerat glesa algoritmer för faltningar (convolutions) som är en viktig operation i många maskininlärningsmodeller för bildtolkning, vilket många föraröververvakningssystem bygger på. Faltningar står ofta för 90-95% av beräkningskraven i sådana nät. Vår kod utnyttjar cachar och vektorenheter och är snabbare än en välimplementerad icke-gles version när 40-50% av vikterna är noll.

De här resultaten öppnar dörren inte bara för resurseffektivare föraröververvakningssystem utan även för förbättringar för andra tillämpningar som behöver bildtolkning baserat på faltningsnät. Det innefattar till exempel system som tolkar trafikskyltar, till exempel för att hålla reda på hastighetsbegränsningar, eller adaptiv helljusautomatik som bländar av I just den riktning där det till exempel kommer en mötande bil. Beskärning är inte begränsat till faltningar utan kan användas i andra typer av nät som till exempel känner igen tal.

3. Background

Around 90% of road traffic accidents are due to driver errors, and the driver was distracted in two thirds of all accidents. Hence driver monitoring systems are attracting increasing attention, and EU regulations are expected to make such systems mandatory by 2025.

We propose a prestudy on power and cost efficient driver monitoring. Recent developments makes this an especially pertinent issue. The upcoming EU regulations will make driver monitoring a standard feature of vehicles rather than a premium feature that the OEM can charge extra for, driving a requirement for low cost. For an electric vehicle, high power consumption for auxilliary sytems decreases range, driving a requirement for low power.

Driver monitoring is one of several sophisticated support systems enabled by the advances in machine learning. While machine learning is a powerful approach, its high computational resource requirements translate into increased component cost and energy consumption, which effectively cap its full potential for vehicles. Based on our expertise in resource constrained systems and machine learning, we will explore mapping driver monitoring to low cost, low power embedded targets. We expect the results to enable improvements in cost and energy efficiency for driver monitoring systems, and also for similar machine learning based features.

Improvements in resource efficiency can be leveraged for several system-level benefits:

- Improve performance in terms of for instance frame rate possibly giving the system shorter reaction time.
- Reduce the resource demands for the system to enable use of lower cost and lower power hardware.
- Allow co-location (consolidation) with other functions on the same hardware, allowing for reduction in component numbers and cost.
- Reduce cost by allowing for lower cost optics by improved signal processing/machine learning models that would otherwise be too computationally expensive.

State of the art in driver monitoring and resource efficient machine learning

There has been a considerable amount of research on driver monitoring, going back at least to the nineties. This early work (see for instance the paper by Knipling and Wierwille [Knipling and Wierwille, 1994]) typically focused on drowsiness and was based on driver inputs, for instance steering, to gauge driver alertness and find early signs of deteriorating performance. Driver inputs continue to be an imortant data source for detecting distractedness [McDonald et al., 2020].

Affordable cameras enabled direct observation of the driver, see for instance the work by Masood and others [Masood et al., 2020]. Several attributes have been used, such as head position and orientation, hand position, facial expression. In particular, eye tracking has been used to infer driver distractedness, that is, lack of concentration on the task of driving. Eye closure has been used as a signal of driver drowsiness.

There has been significant industrial interest in using machine learning based on image (video) data for driver monitoring, for instance by Tobii, SmartEye, and others. Visteon has published a white paper [Sancheti et al., 2019] containing a description of a system using multiple deep learning models.

With cameras come the need for image recognition, which is typically provided with some form of deep learning. Convolutional neural networks has become the most common neural network architecture for image recognition in recent years.

A deep learning model consists of a model architecture which determines the number, kind, and size of layers and a set of parameters, which can be large, used by the layers. The values of the parameters are found by off-line training of the model, typically using a set of examples of input and desired output. The parameters and the code to implement the model architecture can then be used with new inputs to produce appropriate output, a process called inference. This pre-study deals with the resources used by inference since only the inference part of the process is performed in the vehicle.

Several methods have been proposed to make deep learning inference more resource efficient (see for instance Goel and others for a recent survey in the field of image processing [Goel et al., 2020]). With quantization, each model parameter is represented using fewer bits, for instance as an 8-bit integer rather than as a 32-bit floating point number, a degree of precision often used for training. Pruning removes unnecessary parameters entirely, yielding a sparse network. Parameters are considered unnecessary if they have a weak influence on the result (typically if they have values close to zero). Other methods include matrix factorization and filter compression, the latter a technique specific to convolutional networks often used in image processing.

One obstacle to wide adoption of pruning is the need for special data structures that are not available in common machine learning libraries.

4. Purpose, research questions and method

This pre-study intends to identify promising dirctions for improving the resource efficiency of driver monitoring systems through a vertically integrated approach. The implementation of machine learning models in driver monitoring systems was our focus, in particular inference using deep learning models.

Specifically, we identified pruning as a potential source of improved computational efficiency. Therefore, our research question (in the context of this pre-study) became:

To what extent can the results of pruning be exploited to improve the computational efficiency of driver monitoring?

We addressed this question by investigating the computationally most significant operation used by deep neural networks to solve image processing problems, the 2d-convolution. We estimate that for a conventional implementation, 90-95% of the execution time is spent in convolution operations.

A convolution is parameterized by a large number of weights, up to several millions in popular deep neural networks. Pruning is a set of methods for training networks where a large fraction of these weights are zero. In principle, operations on these weights can be eliminated, saving computational resources. Such algorithms are known as sparse algorithms. In practice, sparse computation is nontrivial and conflicts with exploiting features of modern CPUs such as SIMD vector units and caches since skipping zeroes makes the computation less regular.

Because of this conflict, current off-the-shelf machine learning frameworks provide little support for sparse computation, which led us to design our own implementations of sparse convolutions.

Since the motivation for our work is to improve driver monitoring systems, we have chosen to target the ARM architecture since it is often used for embedded systems within the automotive industry. While the general principles of sparse algorithms are quite machine independent, implementing code that efficiently uses hardware features like SIMD vector units and caches (as the conventional dense implementations do) requires a target aware approach.

To get a baseline for comparison, we also implemented dense (that is, not sparse) convolutions tuned for the same hardware.

5. Objective

The objective of this pre-study was to identify challenges in the resource efficient implementation of driver monitoring systems, and also identify some appropriate methods to meet these challenges.
During the work we have focused mostly on exploiting pruning for deep convolutional neural networks as this appeared to to be the most promising approach.

6. Results and deliverables

We have implemented and evaluated a set of sparse algorithms for exploiting the opportunities for reduced execution time offered by unstructured pruning. Our results indicate that there is indeed a potential for improving execution time with pruning. This will lead to lower cost for implementation of driver monitoring, which was the driver of the current work. But the significance of the results goes much further. There are several other safety enhancing driver assistance systems that are or could be implemented using deep learning. Examples include:

- Road sign reading systems, for instance to keep track of speed limits or other restrictions such as overtaking bans. Basing these on image recognition makes them independent of centralized data bases which should be beneficial for instance for temporary changes caused by road work.
- High beam assist systems that identify when there is risk of dazzling oncoming vehicles or other road users. For a system controlling a matrix led light source, it is importand to identify which part of the light cone that need to be switched off. Preferably, such a system should switch off only where there is a risk of dazzling and not for other light sources.
- While the focus of this work was convolutional neural networks, pruning is applicable to any style of neural network. Recurrent neural networks have for instance been used for voice control systems and other speech-to-text applications that have become increasingly common in the automotive field.

Given that safety enhancing systems over time tend to migrate from expensive optional feature to common or even mandated standard equipment, reductions in implementation cost become even more crucial.
We judge that we have met the goals of the project by identifying a promising approach to improving the cost efficiency of driver monitoring systems based on deep learning by realizing the potential of pruning using sparse convolutions.
This plan for the pre-study includes one deliverable in the form of a report summarizing our findings. In line with our decision to focus on the exploitation of pruning, that is also the subject of the report, entiteled “Sparse Convolutions for Pruned ML Models”. The following sub section is a brief summary of that report.

Pruning and sparse convolutions

Pruning replaces some weights with zeroes so that the filters that are used in inference contain a large fraction of zeroes. This property of the filters can in principle be exploited to reduce the number of arithmetic operations that need to be performed. The filter weights are essentially used in inner product operations with data from activations derived from the input images, and since multiplication by zero yields zero, these multiplications can be avoided for zero weights. Similarly, the multiplcation results are summed to produce the output of the convolution, and adding zero can also be avoided.

A sparse algorithm realizes these potential benefits by only performing the computations assiciated with non-zero weights. While this saves arithmetic operations, it reduces the regularity of the computation, making it more difficult to exploit microarchitecture features like SIMD vector operations (known as SSE and AVX in the x86 sphere and Neon in the context of ARM).

The focus of this pre-study was to identify promising approaches to improving the computational efficiency of driver monitoring systems. We chose to focus on exploiting pruning since Tobii had previously studied the application of pruning in their problem domain but had lacked the well-tuned sparse implementations to fully exploit the benefits.

The following figures show the performance results from our experiments. We have compared our sparse implementations with conventional dense implementations of the same filter dimensions. The sparse execution time varies with the fraction of zero weights while the execution time of the dense implementation does not.

In all cases, we present the normalized execution time, that is, the sparse time for a certain configuration divided by the dense time for all configurations. Thus a relative execution time of one means that the sparse and dense implementations were equally fast.

Looking at the results in figures 1-5, we see that when we have no zero weights, the sparse implementation is slower than the dense one. The difference varies from an extra 40% execution time to almost double time. This is expected and is a consequence of the overhead of the sparse algorithms. We also see that the sparse execution time decreases almost linearly with decreasing fraction of non-zero weights (increasing sparseness). Taken together, these two effects imply that the balance point where sparse methods outperform dense ones occurs at about 40-50% zeros depending on configuration.

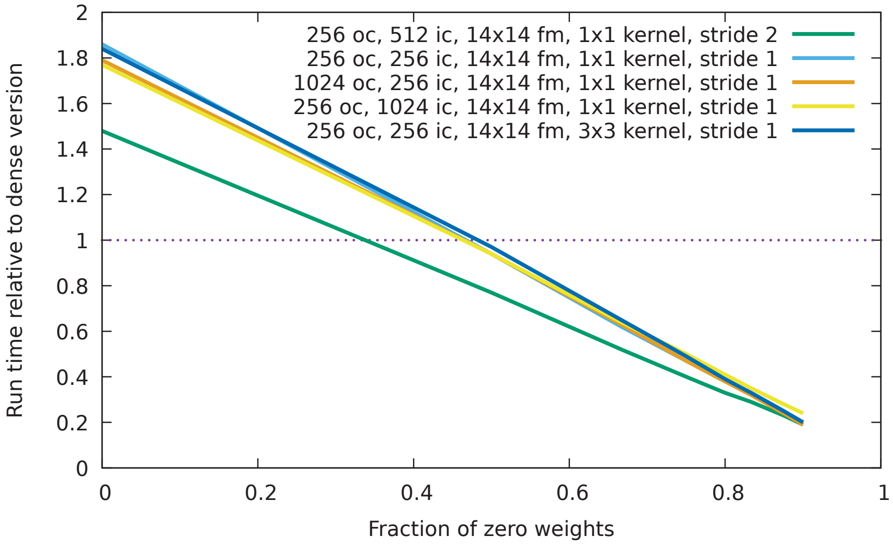

Figure 1: Execution time relative to dense implementation for convolutions with feature maps of size 14x14 for different kernel sizes and numbers of input and output channels

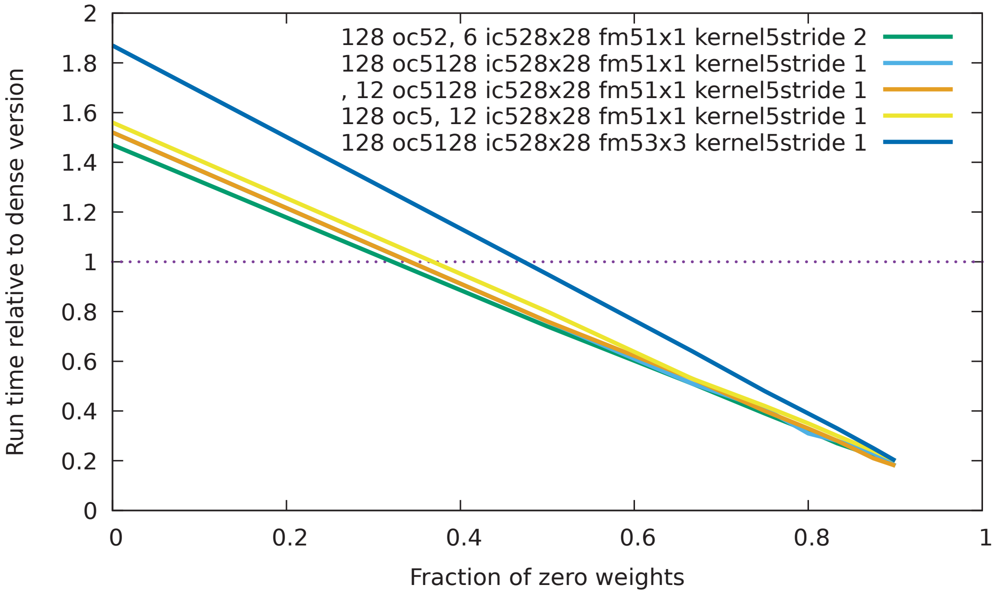

Figure 2: Execution time relative to dense implementation for convolutions with feature maps of size 28x28 for different kernel sizes and numbers of input and output channels

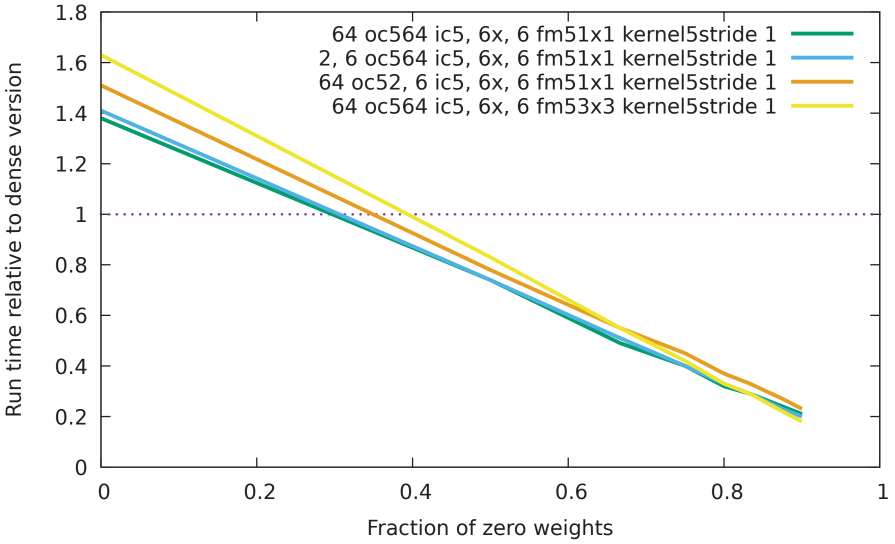

Figure 3: Execution time relative to dense implementation for convolutions with feature maps of size 56x56 for different kernel sizes and numbers of input and output channels

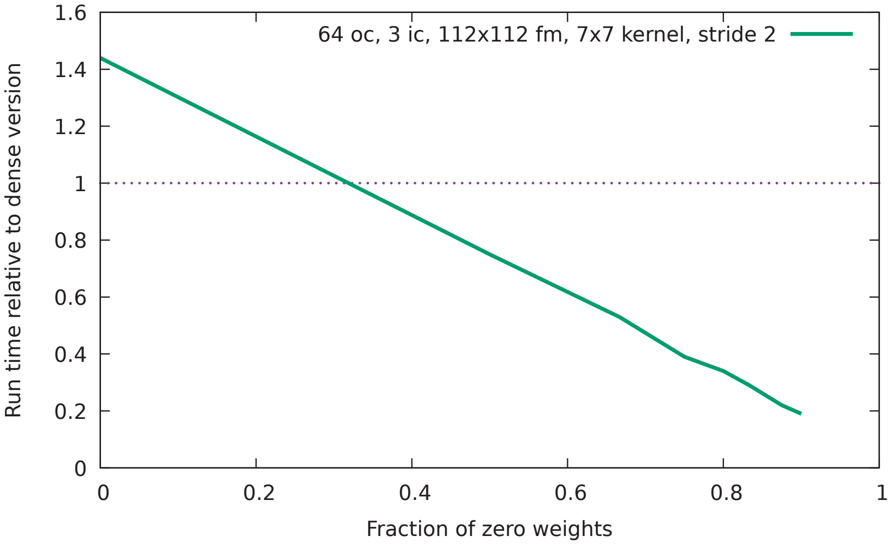

Figure 4: Execution time relative to dense implementation for convolutions with feature maps of size 112x112 for different kernel sizes and numbers of input and output channels

7. Dissemination and publications

7.1 Dissemination

How are the project results planned to be used and disseminated?	Mark with X	Comment
Increase knowledge in the field	X	
Be passed on to other advanced technological development projects	X	We are planning to submit a proposal for a full project in the March 2023 call.
Be passed on to product development projects		
Introduced on the market		
Used in investigations / regulatory / licensing / political decisions		

7.2 Publications

We plan to publish the report on sparse convolutions as a technical report.

8. Conclusions and future research

We have investigated opportunities for improving driver monitoring systems based on deep convolutional neural networks. We have focused on exploiting the potential for increasing resource efficiency offered by pruning. Our results show that compute times can be reduced by a factor of two or more by using data structures and algorithms for sparse computations when 25% of the weights are non zero.

Now that we have demonstrated the potential performance improvements, there are several additional avenues to explore:

- Combining pruning and sparseness with using narrower data types such as Bfloat16 and fixpoint numbers.
- Exploring the precision-performance trade-offs for pruning in the context of the application to driver monitoring.
- Evolving our core algorithms into a tool that could be used more widely in and beyond the automotive industry.
- Explore further applications within the automotive industry.

9. Participating parties and contact persons

This work was performed in collaboration between RISE, Research Institutes of Sweden, AB and Tobii AB.
RISE Research Institutes of Sweden is Sweden's research institute and innovation partner. Through international collaboration with industry, academia and the public sector, we ensure business competitiveness and contribute to a sustainable society.
Tobii is the world leader in eye tracking in terms of overall market share, technology and patent portfolio. Tobii's mission is to create the conditions for new insights into human behavior and intuitive user interfaces with eye tracking. The company was founded in Sweden in 2001 and is today a global company with offices located in Asia, Europe, North America, and South America, supported by a global network of resellers.

Contact:

- At RISE, Karl-Filip Faxén, `karl-filip.faxen@ri.se`
- At Tobii, Mark Ryan, `Mark.Ryan@tobii.com`

References

A. Goel, C. Tung, Y. -H. Lu and G. K. Thiruvathukal, *A Survey of Methods for Low-Power Deep Learning and Computer Vision*, 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), 2020.

R.R. Knipling and W.W. Wierwille: *Vehicle-Based Drowsy Driver Detection: Current Status and Future Prospects*, paper delivered at the IVHS America Fourth Annual Meeting, Atlanta, GA, April 17-20, 1994.

Sarfaraz Masood, Abhinav Rai, Aakash Aggarwal, M.N. Doja, Musheer Ahmad: *Detecting distraction of drivers using Convolutional Neural Network*, Pattern Recognition Letters, Volume 139, 2020, Pages 79-85.

McDonald AD, Ferris TK, Wiener TA. Classification of Driver Distraction: A Comprehensive Analysis of Feature Generation, Machine Learning, and Input Measures. *Human Factors*. 2020;62(6):1019-1035. doi:10.1177/0018720819856454

N.K. Sancheti, K.H. Gopal, and M. Srikant, Camera based driver monitoring system using deep learning, white paper from Visteon, 2019, URL: https://www.visteon.com/wp-content/uploads/2019/04/camera-based-driver-monitoring-system-using-deep-learning.pdf