Quantised Siamese Neural Networks

for Energy Efficient Real-Time Object Tracking

Dominika Przewlocka-Rus

Mateusz Wasala & Hubert Szolc & Krzysztof Blachut & Tomasz Kryjak

Embedded Vision Group, Computer Vision Laboratory,

AGH University of Science and Technology, Krakow, Poland

{dominika.przewlocka, wasala, szolc, kblachut, tomasz.kryjak}@agh.edu.pl

Abstract:
Object tracking is a complex problem, useful in many applications, such as autonomous vehicles or widely understood video surveillance systems. On top of the need for high accuracy and low latency, the requirement of energy efficiency is becoming increasingly important. Following the results of Visual Object Tracking Challenges from previous years, the state-of-the-art trackers tend to be usually based on deep learning. However, because of the computational and memory complexity of these algorithms, achieving real-time processing requires parallelising the computations and dedicated hardware. To meet the energy efficiency requirements, the acceleration on high-end GPUs is not sufficient. Therefore, we propose an optimisation of a visual object tracking approach using Siamese neural networks for embedded, low-power vision systems. We have reduced the network complexity by quantising weights and activations. A number of training scenarios were tested with varying levels of optimisation and their influence on the tracking performance was evaluated. The obtained results indicate that using quantisation can significantly reduce the memory and computational complexity of the proposed network while still enabling precise tracking, thus allowing use in embedded vision systems.

The work presented in this poster was supported by the National Science Center project no. 2016/23/D/ST6/01389

Motivation & Tracker Overview

In the presented research, we focus on adapting a state-of-the-art tracker based on Siamese Neural Networks to embedded devices. We aim to optimise the network for memory and computational complexity which shall directly translate into meeting real-time requirements via an FPGA implementation. A Siamese network is a Y-shaped network with two branches joined to produce a single output. Irrespective of the branches' structure, a Siamese network can be considered as a similarity function that measures the resemblance between two inputs. We can define the basic Siamese architecture as $y = \gamma(\phi(z), \phi(x))$, where ϕ stands for extracting deep features (e.g. via network's branches), z represents the template (e.g. exemplar, object to track), x represents the patch that is compared to the exemplar (e.g. search region, ROI – region of interest) and γ is a cross-correlation.

In this research, we focus on the tracker presented initially in the paper [1] and then modified in the work [2], mainly due to its good performance and simplicity (which is important in the context of FPGA implementation). Its scheme is presented in Fig. 1. The input z represents the tracked object selected in the first frame and scaled to 127x127 pixels. In the next frames, the ROI around the previous object's location is selected and presented to the network as the input x, scaled to 255x255 pixels. Both patches (exemplar and ROI) result in two blocks of feature maps of sizes 17x17x32 and 49x49x32 for z and x respectively. Finally they are cross-correlated and a heat map of size 33x33 is obtained – its highest peak indicates the object's location. The ROI is selected approximately 4 times greater than the object's size, several scaled versions of the ROI are processed to handle object's size variations.

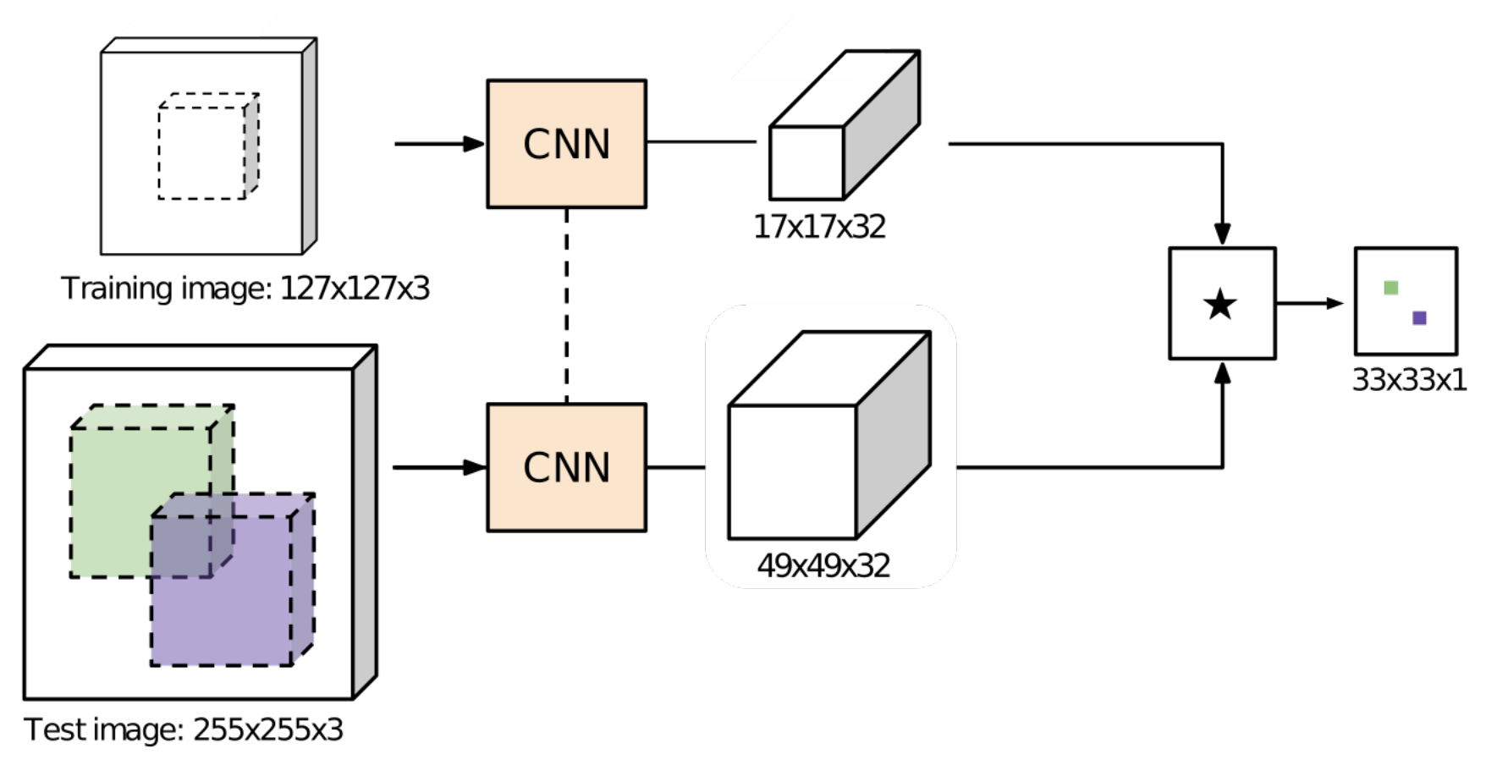

Figure 1: A fully convolutional Siamese Network for tracking [1].

Methods & Tools

To implement deep learning algorithms in FPGA devices, it is highly advisable to optimise the network's architecture. This increases the performance and, in some cases, even allows to only use internal memory resources. Apart from designing lightweight networks, there are several other options to compress it's parameters, like quantisation and pruning. We have used the Brevitas library for Pytorch, that enables the training of quantised networks.The trained model can be then compiled into a hardware design using the FINN tool. Quantisation allows to reduce the number of bits needed to store values, which can be (and often is) critical for deploying networks on embedded devices. Both weights and activations can be quantised. Despite several approaches involving quantising weights after training, the best results are obtained when training the network with reduced precision parameters. Using Brevitas, one can set both the type of quantisation (uniform integer quantisation, binary or ternary), as well as a particular bit width for weights and activations.

Experiments

To obtain representative and comparable to each other results, we have defined the same training parameters for all test cases. The networks were trained using the ILSVRC15 dataset. For the presented experiments, approximately 8% (that is, 10000 image pairs of object and ROI) of the dataset was used for training. The authors of the original Siamese Network compared the accuracy – measured as the average IOU – of their tracker while training on different portions of the ImageNet dataset. In that case, the 8% evaluation resulted in an 48.4% accuracy (for the whole dataset: 52.4%).

No	First Layer	Hidden Layers	Last Layer	Activation	Conv Parameters
1	FP 32	FP 32	FP 32	FP 32	85.5 Mb
2	INT 16	INT 16	INT 16	INT 32	38.3 Mb
3	INT 16	INT 4	INT 8	INT 32	13.4 Mb
4	INT 16	TERNARY	INT 8	INT 32	9.3 Mb
5	INT 16	BINARY	INT 8	INT 32	7.2 Mb

Table 1: Quantisations of the proposed Siamese Neural Network for tracking. FP stands for floating point precision, INT for uniform integer quantisation, TERNARY and BINARY respectively, for two and one bits precision.

The conducted experiments are summarised in Table 1. They were selected to evaluate how the reduced precision affects the training progress and it's results. The first case represents the network with floating point precision and is treated as the baseline. The last column shows the memory needed for convolutional filters (other parameters are constant for all test cases). It is worth noting that on FPGA devices the use of floating point computations is generally more complex, resource consuming, and thus less energy efficient than fixed-point ones.

Results

Network	Precision [%]	IOU [%]
FP	38.58	28.42
INT16	42.87	31.24
INT4	36.73	27.83
TERNARY	44.35	33.40
BINARY	44.22	32.59

Table 2: Average tracking precision and IOU.

The tracking results are presented in Table 2. The tracker was evaluated on a dataset constructed from TempleColor, VOT14, VOT16 sets (without OTB sequences – same as in the original paper). The tracking precision is calculated as the ratio of the number of frames with centre error below the predetermined threshold (in our case equal to 20) in relation to the length of the given sequence. The analysis of the average precision values allows to draw interesting conclusions, while comparing the quantised versions to the baseline floating point one. First of all, there is a precision growth with increasing quantisation (excluding the INT16 network), with the binary network achieving the precision of 44.22 on the top. A similar tendency can be observed using the IOU (Intersection Over Union) metric. The obtained results are not as good as in the original paper, nevertheless this situation should be solved while training the network for a longer time (as planned in the future work). It is essential to notice that the binary tracker is relatively better than the one using floating point computations.

However, if we acknowledge that neural networks are generally redundant structures, this behaviour is not so astounding, as by quantising weights, the overfitting is prevented. Regardless the lower memory requirements, such quantisations could have impact on the inference time. Given ternary or binary networks, the computational complexity is reduced via replacing standard MAC (Multiply and Accumulate) operations with their binary equivalents. Such operations, if properly implemented, lead to lower execution time. In the case of FPGA implementation, it also results in lower resource usage.

References

[1] Luca Bertinetto, Jack Valmadre, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. Fully-convolutional siamese networks for object tracking. *CoRR*, abs/1606.09549, 2016.

[2] Jack Valmadre, Luca Bertinetto, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. End-to-end representation learning for correlation filter based tracking. *CoRR*, abs/1704.06036, 2017.