DEF: Differential Encoding of Featuremaps for Low Power Convolutional Neural Network Accelerators

Alexander Montgomerie-Corcoran
Imperial College London
London, UK
alexander.montgomerie-corcoran15@imperial.ac.uk

Christos Savvas-Bouganis
Imperial College London
London, UK
christos-savvas.bouganis@imperial.ac.uk

ABSTRACT

As the need for the deployment of Deep Learning applications on edge-based devices becomes ever increasingly prominent, power consumption starts to become a limiting factor on the performance that can be achieved by the computational platforms. A significant source of power consumption for these edge-based machine learning accelerators is off-chip memory transactions. In the case of Convolutional Neural Network (CNN) workloads, a predominant workload in deep learning applications, those memory transactions are typically attributed to the store and recall of featuremaps. There is therefore a need to explicitly reduce the power dissipation of these transactions whilst minimising any overheads needed to do so. In this work, a Differential Encoding of Featuremaps (DEF) scheme is proposed, which aims at minimising activity on the memory data bus, specifically for CNN workloads. The coding scheme uses domain specific knowledge, exploiting statistics of the featuremaps alongside knowledge of the data types commonly used in machine learning accelerators as a means of reducing power consumption. DEF is able to out-perform recent state-of-the-art coding schemes, with significantly less overhead, achieving up to 50% reduction of activity across a number of modern CNNs.

1 INTRODUCTION

Power-efficiency is a sought after characteristic in many CNN accelerators. As the use-case for these accelerators moves closer to edge-based applications, the constraints of the platform they are deployed on become more and more prominent. In many applications the power-budget of the system is the main limiting-factor, which often places a bottleneck on the computational performance of the CNN accelerator. Previous work identified that accessing data stored in the DDR memory is a significant source of power consumption for many embedded systems [6, 10]. The power consumption of the memory subsystem is due to both the DDR chip itself as well as due to the bus-lines connecting the DDR to the device. These bus-lines have high capacitance compared to on-chip routing, and therefore they often consume a considerable proportion of the total dynamic power of the system. Dynamic power consumption for a bus is modelled by Equation (1):

$$P_{dynamic} = n \cdot V_{dd}^2 \cdot f_{clk} \cdot C \cdot a \tag{1}$$

where n is the number of bus lines, V_{dd} is the driving voltage, f_{clk} is the clock frequency of the bus, C is the capacitance of the bus lines and a the the activity on the bus. The values n, V_{dd}, f_{clk} and C are system parameters that are fixed for a specific platform. The activity, a, however is highly dependant on the workload and how it is encoded for transmission.

As the activity is dependant on the memory transactions, a common technique to address high activity is to introduce a coding scheme for reducing the bus activity, both for address lines and data lines of the bus. Existing coding schemes introduce either *spatial* [9, 18] or *temporal* [17] redundancy in order to carry information about the coding scheme. Temporal redundancy introduces extra words into the data stream, impacting performance. Spatial redundancy introduces extra bus-lines, constraining the available resources. Moreover, coding schemes can be categorised as either *adaptive* [9, 17, 18] or *static* [12]. Static coding schemes utilise fixed properties of the bus activity to design their coding scheme, resulting in approaches with the need of limited resources, whereas adaptive schemes use online methods.

The coding scheme proposed in this work is a static scheme that utilises knowledge of the CNN workload in order to efficiently reduce the activity along the off-chip memory bus. By using statistical knowledge of the featuremaps transferred between the memory and the accelerator during execution of a CNN workload, the coding scheme is able to achieve improved reduction in activity compared to recent state-of-the-art coding schemes. This is achieved without introducing any temporal or spatial redundancy.

The paper is structured as follows: Section 2 covers background work on CNN accelerators and their workloads. Related state-of-the-art coding schemes are also introduced. Section 3 introduces DEF, the proposed coding scheme. Finally, the proposed coding scheme is evaluated against other coding schemes and results obtained by its prototyping on an FPGA device are reported in Section 4.

2 BACKGROUND & RELATED WORK

2.1 Activity Minimisation Schemes

There have been many coding schemes proposed in order to reduce activity on the off-chip busses. The main activity reduction schemes of interest are Bus-Invert (**BI**) coding [18], Probability-Based Mapping (**PBM**) [8, 12], Adaptive Bus Encoding (**ABE**) [17] and Adaptive Word Reordering (**AWR**) [9]. BI minimises the maximum number of transitions. The inverse of the incoming signal is sent if the hamming distance with the previous signal is greater than $\frac{n}{2}$. An extra bus line is needed to indicate this transition. PBM combines an entropy encoder with a bitwise decorrelator[19]. This ensures that the most frequent words create the least transitions. ABE finds clusters of highly correlated bus-lines within a window and performs an *XOR* operation with the basis line and the cluster. An extra bus line and word are required to propagate the basis information over the channel. AWR reorders the incoming words such that there are fewer transitions between neighbouring words, using a nearest neighbour algorithm. Extra bus-lines are required to recover the original ordering.

2.2 CNN Accelerators

A number of edge-based accelerators have been proposed for CNNs [23]. Typically, these accelerators are either streaming-architectures [1, 21] or single computation engines [2, 4, 14]. The constraints of edge-devices has made it difficult to faithfully implement CNN networks on them due to the large number of parameters and costly implementation of floating point arithmetic. As such, research has focused in reducing the size and computational complexity of CNN networks. Commonly used techniques are based on the quantisation of featuremaps/weights as well as in the pruning of the network's parameters [5, 24]. Quantization schemes are employed to reduce the number of bits used to represent weights and feature maps, as well as reducing the computational cost of the arithmetic used in the CNN, with fixed-point becoming the de-facto representation in edge-tailored CNN accelerators [1, 2, 22]. The bit-width of these representations generally range between 1 to 16 bits.

Moreover, a number of proposed CNN accelerators identify off-chip memory as a significant source of power dissipation, with representative examples being the EYERISS [2] and the eCNN [7] accelerators. The EYERISS accelerator employs Run Length Encoding (RLE) between off-chip memory and the accelerator hardware. It makes use of the abundance of zeros found in featuremaps, allowing for significant compression to be achieved, reducing the number of memory transactions. In eCNN, a Huffman coding scheme for parameters is introduced in order to reduce the memory transactions. Additionally, power-awareness has been explored in the context of CNN accelerators in [10], where power consumption was modelled for the *fpgaConvNet* [21] accelerator, which also encapsulated power consumption of off-chip memory. By using this model, a design space exploration tool was able to discover power-efficient designs for given performance targets.

The proposed methodology either departs from the problem setting of the above approaches (in the case of activity minimisation schemes) or provides improved trade-offs between power-consumption reduction and necessary resources for its implementation. The above are achieved by providing a domain specific off-chip bus activity reduction approach capitalising on the statistical properties and access patterns that are commonly seen in CNN workloads.

3 PROPOSED CODING SCHEME

The proposed coding scheme aims at reducing the dynamic power consumption by reducing the activity along the data bus connected to off-chip memory. The average activity, denoted by a and is given by Equation (2) for a bus width of n bits and transmitting a stream, $\boldsymbol{x}$, of m words.

$$a(\boldsymbol{x}) = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=0}^{n} x_{i,j} \oplus x_{i,j-1} \tag{2}$$

where $x_{i,j}$ is the j^{th} bit of the i^{th} word in the stream.

Following a similar approach as many state-of-the art coding schemes [8, 12, 13], a decorrelator is introduced into the encoding path of the scheme. The decorrelating function is described in Equation (3) for the i^{th} word in the stream.

$$d(x_i) = x_i \oplus d(x_{i-1}) \tag{3}$$

Introducing the decorrelator into the encoding path means that the activity of a decorrelated stream can now be viewed as is described in Equation (4).

$$\hat{a}(\boldsymbol{x}) = a(d(\boldsymbol{x})) = \frac{1}{m \cdot n} \sum_{i=0}^{m} \sum_{j=0}^{n} x_{i,j} \tag{4}$$

This view of activity reveals a far simpler objective of the coding scheme: rather than minimising the number of transitions between two consecutive bits, instead the popcount (number of "1" bits) per word needs to be minimised. This new look on activity suggests a minimisation objective for $x_{i,j}$, however the bit-wise view of the bus does not immediately reveal any arithmetic solution.

To achieve a reduction in total "1" bits, the first step taken in the proposed coding scheme is to consider the representation of data being sent along the bus. As quantization of featuremaps is becoming increasingly more common for CNN accelerators, the proposed scheme makes the assumption that the data transmitted across the bus is an n-bit fixed point integer. Twos complement (TC) is a common integer representation however the redundancy of repeated sign bits adds significantly to the activity. Therefore a sign-magnitude (SM) integer representation is adopted [11]. It is worth noting that the full range of the TC value can be preserved in the SM representation, by keeping the value of -2^{n-1} the same in both representations. With the SM representation, the popcount for each integer value is distributed more symmetrically around 0. This point is illustrated in Figure 1, where the symmetry can clearly be seen.

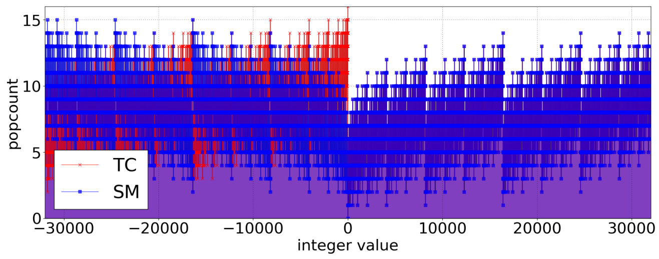


Figure 1: Popcount for all 16-bit signed integer values for both TC and SM representation.

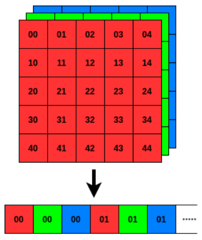
Using this representation allows for certain properties regarding the activity to be guaranteed, namely the following:

Theorem 1. *Let $\boldsymbol{x}$ be a stream of n-bit twos complement integers with m elements. Let $\boldsymbol{y}$ be the respective sign-magnitude integer representation of $\boldsymbol{x}$. If, $|x_i| < 2^k \; \forall \; i \in [0, m]$, and $k < n$, then the activity of $\boldsymbol{y}$ after it is decorrelated is bounded such that $\hat{a}(\boldsymbol{y}) \leq \frac{k+1}{n}$.*

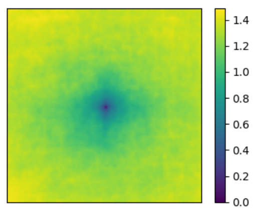
Proof. For a sign-magnitude integer, y, if the absolute value is below a threshold, that is $|y| < 2^k$, then the *popcount* is such that $\sum_{j=0}^{n} y_{i,j} \leq k+1$. If this bound is held for all words in a stream, $\boldsymbol{y}$ of sign-magnitude integers, then the activity of the stream is bounded such that, $\hat{a}(\boldsymbol{y}) = \frac{1}{m \cdot n} \sum_{i=0}^{m} \sum_{j=0}^{n} y_{i,j} \leq \frac{1}{m \cdot n} \sum_{i=0}^{m} (k+1) \leq \frac{k+1}{n}$ □

As can be seen, a bound on the absolute of the sign-magnitude stream guarantees a bound on the activity also. As such, by decreasing the magnitude of each word in the stream, the activity is also decreased. This observation leads to the objective of the differential encoding component.

To begin to tackle this objective, attention is now given to the machine learning domain. Featuremaps produced by CNN workloads exhibit similarity in neighbouring pixels. This similarity leads to a low absolute difference, as is illustrated in Figure 2b.



(a) Channel-first streaming.



(b) Similarity in ImageNet dataset.

Figure 2: Feature maps in CNN accelerators.

In this figure, the distance between the middle pixel and all other pixels is taken, and averaged across a sample of images from the Imagenet dataset [15]. Only a single channel of the input featuremap is shown. This figure highlights the fact that neighbouring activations of the same channel of a featuremap on average have a small absolute difference. This property is carried throughout the CNN, although becomes less prominent deeper into the network, due to the translation invariance properties of CNNs.

The proposed coding scheme exploits pixel similarity by taking the difference between an activation and it's neighbour of the same channel, as described in Equations (5a) and (5b), for the i^{th} word in the stream:

$$\textit{(diff encoder)}\ \ \hat{x}_i = x_i - x_{i-k} \tag{5a}$$

$$\textit{(diff decoder)}\ \ x_i = \hat{x}_i + x_{i-k} \tag{5b}$$

where k indicates the distance in the stream from the neighbouring activation of the same channel. This value, k, depends on the data-access pattern of the featuremap from the off-chip memory. For both streaming architectures [1, 21] and single computation engines [3], k is chosen to be the number of channels the featuremap has, as channel-first streaming, which is illustrated in Figure 2a, is common in these accelerators.

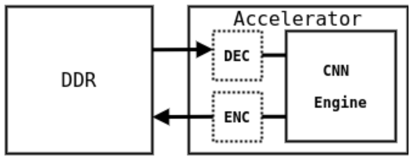
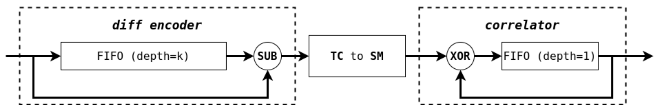
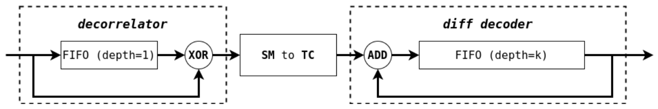


Figure 3: Overview of a system utilising DEF.

The complete system for the DEF coding scheme can be seen in Figure 4 for both the encoder and decoder. This complete coding scheme is lossless, and requires no additional spatial or temporal redundancy. The most significant resource overhead is the buffering needed for the *diff encoder/decoder*, where k words must be stored. The encoder and decoder can be introduced into a CNN accelerator as shown in Figure 3. They are placed in the compute pipeline such that only encoded featuremaps are sent/retrieved to/from DDR.



(a) DEF encoder



(b) DEF decoder

Figure 4: Block diagrams for DEF coding scheme.

4 RESULTS

The proposed coding scheme, *DEF*, is evaluated against existing state-of-the-art coding schemes. These coding schemes are split into activity minimisation schemes (*ABE*, *AWR*, *PBM* and *BI*) and compression schemes (*Huffman* and *RLE*). Firstly, the impact of all the coding schemes on a simulated activity will be evaluated and compared. The impact on power dissipation of off-chip memory will then be compared for an FPGA platform. Stimulus for the coding schemes is generated using the Neural Network Distiller framework [24]. This framework is used to generate quantized featuremaps that are of a signed, asymmetric fixed-point integer type. A random sample of 128 images from the ImageNet dataset [15] are used as input for the quantized networks. A channel-first streaming order is assumed. The metrics of interest for this evaluation are the following:

- *Transitions Ratio*: The ratio of bit transitions ("0" to "1" and "1" to "0") of the encoded stream to the unencoded stream, as given in Equation (6). The transitions relate to the dynamic energy consumed on the bus.

$$t_{ratio} = \frac{\text{Total Transitions}_{\ encoded}}{\text{Total Transitions}_{\ unencoded}} \tag{6}$$

- *Average Activity*: The average number of transitions on the bus per word per bus-line, as described in Equation (7). This metric relates to the average dynamic power consumed by the bus.

$$a_{avg} = \frac{\text{Total Transitions}}{\text{Bus Width} \times \text{Total Words}} \tag{7}$$

4.1 Activity Minimisation Schemes

A comparison is performed between the mentioned activity reduction schemes for different quantization bitwidths for the widely used state-of-the-art network MobileNet v2. This can be seen in Figure 5, where a comparison is made for average activity throughout the MobileNetv2 [16] network. The *BI* and *ABE* coding schemes show lower reduction as the bitwidth is increased, whereas *PBM* improves with increased bitwidth, as well as *DEF* to some extent. *DEF* has the lowest activity across most of bitwidths (apart from a bitwidth of 2). This figure covers the typical range of bitwidths,

Encoding Scheme	t_{ratio}			a_{avg}			**Memory** (B)			**Bus Width** (bits)		
	$n = 4$	$n = 8$	$n = 16$	$n = 4$	$n = 8$	$n = 16$	$n = 4$	$n = 8$	$n = 16$	$n = 4$	$n = 8$	$n = 16$
(unencoded)	-	-	-	0.2897	0.3354	0.3744	0	0	0	4	8	16
ABE	0.9675	1.0028	1.0074	0.2047	0.2832	0.3418	15	31	63	5	9	17
BI	0.8861	0.9155	0.9263	0.1986	0.2671	0.3226	0	0	0	5	9	17
PBM	1.5473	1.1161	0.9245	0.4221	0.3697	0.3499	7	255	131071	4	8	16
AWR	1.3326	1.1350	0.9835	0.2502	0.3012	0.3272	0	0	0	6	10	18
DEF	**0.7182**	**0.7347**	**0.6571**	**0.2043**	**0.2460**	**0.2504**	**174**	**349**	**699**	**4**	**8**	**16**

Table 1: Comparision of metrics and usage of activity minimisation schemes for MobileNetv2.

however 4-bit, 8-bit and 16-bit are the most common throughout accelerators, and so will be the subject of the rest of the evaluation.

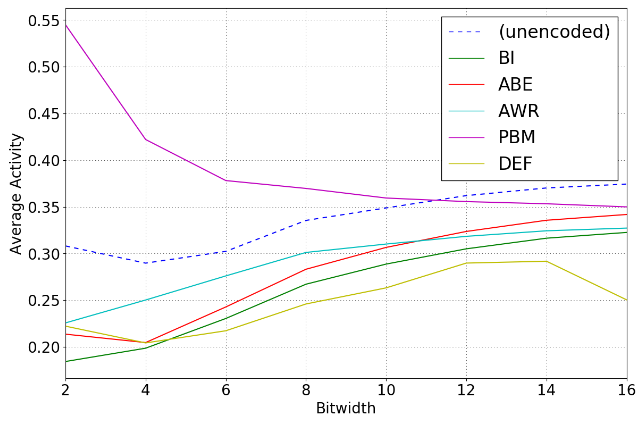


Figure 5: Activity for various bus widths. Evaluated on MobileNetv2.

The activity minimisation schemes are further compared in Table 1 for MobileNetv2. The activity and transition ratio are compared alongside memory usage and bus-width. The memory usage indicates the amount of bytes of storage needed for the implementation of the coding scheme. This table highlights the overheads for each coding scheme and the impact they have on the activity as well as on the number of transitions. *ABE, BI* and *AWR*[1] all require extra bus-lines in order to propagate coding information across the bus. It is worth noting that even a single extra bus-line can become prohibitive for certain platforms due to physical bus-width limits. The impact on extra bus-lines for transitions can also be seen. Both *ABE* and *AWR* generally increase the number of transitions despite a reduction in activity, suggesting that the use of extra bus-lines may have a negative impact on energy consumed. This table highlights the improvements *DEF* brings, where it can be seen to have the lowest transition ratio and activity. This is achieved with no extra bus-lines and minimal impact on resource overheads. Compared to *PBM*, the only other coding scheme which does not require extra bus-lines, *DEF* outperforms on both metrics with comparable or less memory overhead.

To gain insight into where the savings are observed within the CNN, an investigation is made for activity reduction across layers of MobileNetv2 and the results are reported in Figure 6. This figure

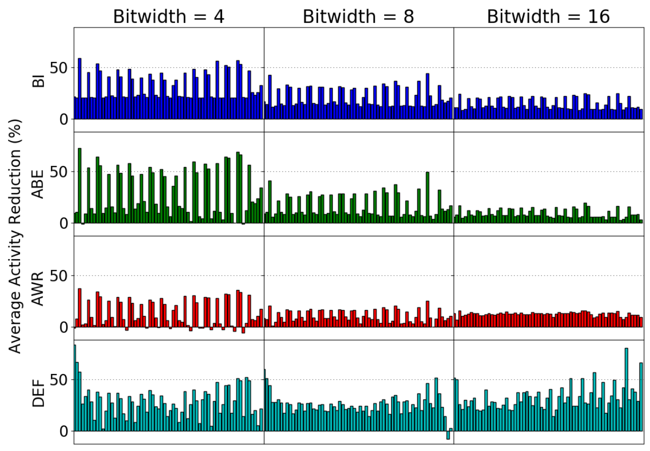


Figure 6: Activity reduction across all layers of MobileNetv2 for bitwidths of 4, 8 and 16.

highlights the variance of activity reduction throughout the different layers and general trends of the coding schemes. It can be seen that all coding schemes have a significant amount of variance at a bitwidth of 4. It can be seen that *DEF* generally has significant reduction for the first few layers, capturing the large pixel similarity observed early on. There is also an increase in activity reduction towards the end, due to the large number of zeros observed in later layers, which the coding scheme is able to exploit.

The activity reduction of the coding schemes is further investigated across seven state-of-the-art CNNs for bitwidths 4, 8 and 16 in Figure 7, where a weighted average is taken across the layers of each network. It can be seen that the reduction is not uniform across CNNs for any given coding scheme or bitwidth. The depth and structure of different CNNs lead to variations in the resulted activity. It can be seen that *DEF* achieves the highest reduction across all networks for bitwidths of 8 and 16, achieving a reduction between 20% and 50%.

4.2 Compression Schemes

The proposed scheme is also compared against popular compression schemes within the ML community. The aim of the comparison is to quantify the extent that compression actually has on dynamic energy consumption, and how this compares to a coding scheme which only targets activity reduction. The compression schemes compared are *RLE* and *Huffman*. Furthermore, the proposed scheme,

[1]The AWR coding scheme is implemented with a window size of 4.

Encoding Scheme	t_{ratio}			a_{avg}			Compression Ratio			Memory (B)		
	$n=4$	$n=8$	$n=16$	$n=4$	$n=8$	$n=16$	$n=4$	$n=8$	$n=16$	$n=4$	$n=8$	$n=16$
(unencoded)	-	-	-	0.1928	0.2470	0.2555	1.00	1.00	1.00	0	0	0
DEF	0.6212	0.6162	0.4789	0.1231	0.1564	0.1261	1.00	1.00	1.00	115	231	462
RLE	2.0020	1.4493	1.2900	0.4465	0.3839	0.3667	1.27	1.17	1.24	0	0	0
DEF+RLE	1.0401	0.7927	0.6034	0.2787	0.2170	0.1914	1.42	1.14	1.34	107	215	393
Huffman	1.3077	1.1605	0.9478	0.4509	0.4876	0.4956	2.00	1.97	2.32	17	388	147111

Table 2: Comparision of metrics and usage of compression schemes for GoogleNet.

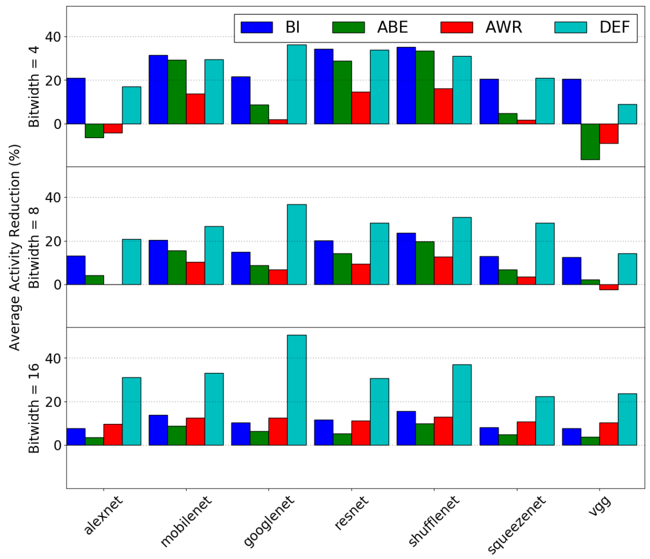


Figure 7: Activity reduction for various CNNs for bitwidths of 4, 8 and 16.

DEF, is also combined with *RLE* to evaluate the impact of the combination of the two schemes on power and energy reduction.

An in-depth comparison is performed between the mentioned approaches using the GoogleNet [20] network, and the results are presented in Table 2, where the compression ratio is also introduced as a metric of interest. The results show that compression has an impact of increasing both activity as well as the number of transitions, as observed in *Huffman* and *RLE* approaches. However, by introducing an activity minimisation scheme into the compression scheme, activity and transition reduction can be achieved, as shown by the performance of the *DEF+RLE* scheme.

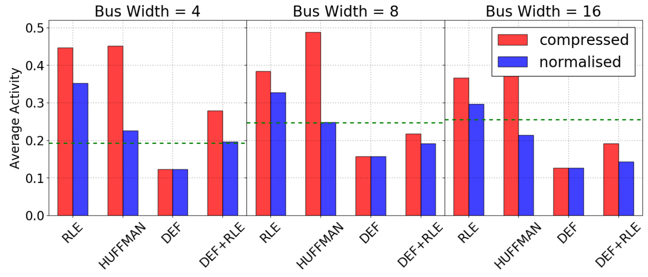


Figure 8: Activity reduction for GoogleNet.

To further evaluate the compression schemes and their impact on activity, a new metric referred to as normalised activity is introduced, and given in Equation (8):

$$\bar{a}_{avg} = \frac{a_{avg}}{\text{Compression Ratio}} \tag{8}$$

This models the activity of the bus if the compressed stream was stretched to an uncompressed length. The activity and normalised activity are compared for the compression schemes as well as *DEF* in Figure 8, for the various bitwidths on GoogleNet. In this figure, the green line represents the baseline activity of the unencoded stream. The results show that *DEF* achieves the lowest activity. The compressed and normalised activity for *Huffman* vary significantly, with the normalised activity placing Huffman close to or below the baseline activity. The combination of *RLE* and *DEF* does not improve on the activity of *DEF* on it's own for either compressed or normalised activity, despite the compression ratio it can achieve.

4.3 FPGA Prototyping

To assess the impact of the proposed coding scheme within a complete CNN accelerator, the power consumption of the memory subsystem supporting a CNN workload is measured on an FPGA platform. A Xilinx ZC702 board is used with a design frequency of 200Hz. The power consumption of the `VCC1V5` rail that drives the IO between the FPGA and DDR as well as the three DDR chips themselves is measured. In the experiment, encoded streams are sent to and from the FPGA, emulating a typical CNN accelerator design which utilises two of the DDR chips independently.

Encoding Scheme	Average Power (mW)
(unencoded)	1556.2
ABE	1522.9
BI	1509.8
PBM	1546.2
AWR	1549.7
DEF	**1458.5**

Table 3: Power consumption in MobileNetv2 for activity minimisation coding schemes on a XC7020.

The power consumption of memory transactions is given for each activity minimisation scheme in Table 3, when a workload from MobileNetv2 is used. It is worth noting that the idle power consumption of the memory sub-system is around 630mW. The results indicate that *DEF* is able to reduce the memory subsystem power by 100mW.

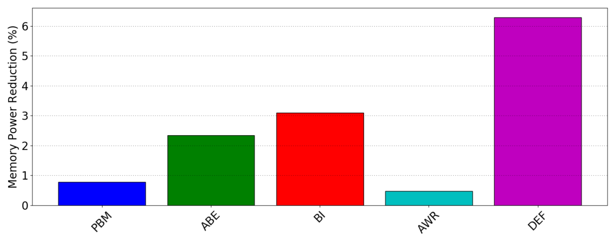


Figure 9: Relative power reduction for MobileNetv2.

The power reduction of the coding schemes is also compared in Figure 9. *DEF* is able to achieve a 6% reduction in power consumption, which is 2× greater than the next comparable coding scheme, *BI*. Please note that reduction in reported power compared to reported activity is lower, as the active power of the DDR chip is also taken into account. The coding schemes mostly impact IO power, and their gains are weighted accordingly to their contribution to the overall active power consumption of the memory subsystem.

The compression coding schemes are also evaluated on the FPGA platform. Power, energy and execution time are given for the first layer of AlexNet in Table 4. The table highlights the significance of the relative power consumption between IO and the DDR chip, as although *DEF* is able to reduce the power to some extent, ultimately *Huffman* is able to transfer the featuremaps more energy-efficiently as the DDR is active for less time.

Encoding Scheme	**Power** *(mW)*	**Time** *(ms)*	**Energy** *(μJ)*
(unencoded)	1534.2	5.519	8466.7
RLE	1526.3	3.833	5849.7
Huffman	1556.8	1.383	2152.9
DEF	1397.7	5.526	7723.8
DEF+RLE	1465.0	2.519	3690.7

Table 4: Power, Execution Time and Energy of compression schemes for AlexNet on a XC7020.

Finally, the overheads of implementing DEF on a XC7020 FPGA are evaluated. The maximum resources needed for MobileNetv2 for 8-bit featuremaps are as follows,

FF :	77,	***LUT***:	344,	***BRAM***:	0,	***DSP***:	0

In total, the featuremap with the greatest number of channels (1280) of MobileNetv2's workload will still only require less than 1% of the total board resources. This means that the power overhead for the implementation will be insignificant also.

5 CONCLUSION

The paper proposed a novel encoding scheme for activity reduction tailored to CNN workloads when communication with off-chip memory is required.The coding scheme is able to achieve up to 50% reduction in activity for featuremaps of CNNs, proving it's suitability for this specific domain. The impact of activity reduction is demonstrated on an FPGA platform, and the coding scheme is able to reduce the power consumption of a memory subsystem by 6%, with minimal overhead.

REFERENCES

[1] Michaela Blott, Thomas B. Preußer, Nicholas J. Fraser, Giulio Gambardella, Kenneth O'brien, Yaman Umuroglu, Miriam Leeser, and Kees Vissers. 2018. FINN-R: An End-to-End Deep-Learning Framework for Fast Exploration of Quantized Neural Networks. *ACM Trans. Reconfigurable Technol. Syst.* (2018).

[2] Y. Chen, J. Emer, and V. Sze. 2016. Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA).*

[3] Y. Guan, H. Liang, N. Xu, W. Wang, S. Shi, X. Chen, G. Sun, W. Zhang, and J. Cong. 2017. FP-DNN: An Automated Framework for Mapping Deep Neural Networks onto FPGAs with RTL-HLS Hybrid Templates. In *2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM).*

[4] Y. Guan, Z. Yuan, G. Sun, and J. Cong. 2017. FPGA-based Accelerator for Long Short-Term Memory Recurrent Neural Networks. In *ASP-DAC.*

[5] P. Gysel, J. Pimentel, M. Motamedi, and S. Ghiasi. 2018. Ristretto: A Framework for Empirical Study of Resource-Efficient Inference in Convolutional Neural Networks. *TNNLS* (2018).

[6] M. Horowitz. 2014. 1.1 Computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC).*

[7] Chao-Tsung Huang, Yu-Chun Ding, Huan-Ching Wang, Chi-Wen Weng, Kai-Ping Lin, Li-Wei Wang, and Li-De Chen. 2019. ECNN: A Block-Based and Highly-Parallel CNN Accelerator for Edge Inference. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture.*

[8] Claudia Kretzschmar, Robert Siegmund, and Dietmar Müller. 2003. Low Power Encoding Techniques for Dynamically Reconfigurable Hardware. *The Journal of Supercomputing* (2003).

[9] E. Maragkoudaki, P. Mroszczyk, and V. F. Pavlidis. 2019. Adaptive Word Reordering for Low-Power Inter-Chip Communication. In *2019 Design, Automation Test in Europe Conference Exhibition (DATE).*

[10] A. Montgomerie-Corcoran, S. I. Venieris, and C. Bouganis. 2019. Power-Aware FPGA Mapping of Convolutional Neural Networks. In *2019 International Conference on Field-Programmable Technology (ICFPT).*

[11] S. Ramprasad, N. R. Shanbha, and I. N. Hajj. 1997. Analytical estimation of signal transition activity from word-level statistics. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (1997).

[12] S. Ramprasad, N. R. Shanbhag, and I. N. Hajj. 1999. A coding framework for low-power address and data busses. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (1999).

[13] S. Ramprasad, N. R. Shanbhag, and I. N. Hajj. 1999. Information-theoretic bounds on average signal transition activity [VLSI systems]. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (1999).

[14] B. Reagen et al. 2016. Minerva: Enabling Low-Power, Highly-Accurate Deep Neural Network Accelerators. In *ISCA.*

[15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* (2015).

[16] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *CVPR.*

[17] S. Sarkar, A. Biswas, A. S. Dhar, and R. M. Rao. 2017. Adaptive Bus Encoding for Transition Reduction on Off-Chip Buses With Dynamically Varying Switching Characteristics. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2017).

[18] M. R. Stan and W. P. Burleson. 1995. Bus-invert coding for low-power I/O. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (1995).

[19] M. R. Stan and W. P. Burleson. 1997. Low-power encodings for global communication in CMOS VLSI. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 5, 4 (Dec 1997), 444–455. https://doi.org/10.1109/92.645071

[20] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015.

Going Deeper with Convolutions. In *Computer Vision and Pattern Recognition (CVPR)*.

[21] S. I. Venieris and C. Bouganis. 2017. Latency-Driven Design for FPGA-based Convolutional Neural Networks. In *FPL*.

[22] S. I. Venieris and C. Bouganis. 2018. fpgaConvNet: Mapping Regular and Irregular Convolutional Neural Networks on FPGAs. *TNNLS* (2018).

[23] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen. 2020. Convergence of Edge Computing and Deep Learning: A Comprehensive Survey. *IEEE Communications Surveys Tutorials* (2020).

[24] Neta Zmora, Guy Jacob, Lev Zlotnik, Bar Elharar, and Gal Novik. 2019. Neural Network Distiller: A Python Package For DNN Compression Research. (2019). https://arxiv.org/abs/1910.12232