Energy Complexity of Fully-Connected Layers

Jiří Šíma[1(✉)] and Jérémie Cabessa[1,2]

[1] Institute of Computer Science of the Czech Academy of Sciences, Prague, Czechia
sima@cs.cas.cz
[2] DAVID Laboratory, UVSQ – University Paris-Saclay, Versailles, France
jeremie.cabessa@uvsq.fr

Abstract. The energy efficiency of processing convolutional neural networks (CNNs) is crucial for their deployment on low-power mobile devices. In our previous work, a simplified theoretical hardware-independent model of energy complexity for CNNs has been introduced. This model has been experimentally shown to asymptotically fit the power consumption estimates of CNN hardware implementations on different platforms. Here, we pursue the study of this model from a theoretically perspective in the context of fully-connected layers. We present two dataflows and compute their associated energy costs to obtain upper bounds on the optimal energy. Using the weak duality theorem, we further prove a matching lower bound when the buffer memory is divided into two fixed parts for inputs and outputs. The optimal energy complexity for fully-connected layers in the case of partitioned buffer ensues. These results are intended to be generalized to the case of convolutional layers.

Keywords: Convolutional neural networks · Energy complexity · Dataflow

1 Energy Complexity Model for CNNs

Deep neural networks (DNNs) represent a cutting-edge machine learning technology, with countless applications in computer vision, natural language processing, robotics, etc. These models are typically composed of hundreds of thousands of neurons and tens of millions of weights, and are thus computationally demanding and highly energy-consuming. With the ever-growing use of mobile devices, like smartphones or smartwatches, comes the issue of the implementation, deployment, and portability of already trained DNNs on low-power hardware. Recently, extensive research has been conducted on techniques that enable energy-efficient DNN processing on a variety of hardware platforms and architectures (e.g., GPUs, FPGAs [4], memory hierarchies) [8]. The proposed techniques reduce the computational cost via hardware design (including massive parallelism) and/or approximation of DNN models. For example, in error-tolerant applications such as image classification, the use of approximate computing methods [3] (e.g. low

I. Rojas et al. (Eds.): IWANN 2023, LNCS 14134, pp. 3–15, 2023.
https://doi.org/10.1007/978-3-031-43085-5_1

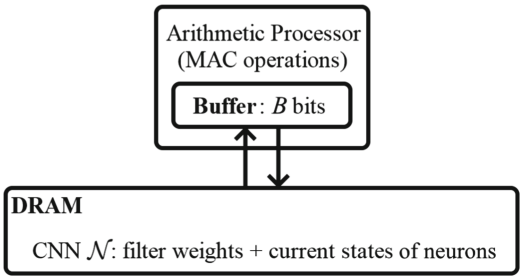

Fig. 1. The energy complexity model.

float precision, approximate multipliers) can save an enormous amount of energy at the cost of only a small loss in accuracy.

For a particular DNN hardware implementation, the power consumption of the inference process can be either practically measured or analytically estimated using physical laws. This power consumption depends on parameters and constants related to the hardware architecture, and hence, its evaluation varies for different hardware implementations. Some computer programs [5,9] can optimize the power consumption of a particular DNN on various hardware platforms [2,6]. It has been empirically observed that the energy cost of DNN processing mainly consists of two components: the computation energy, and the data energy which represents around 70% of the total cost [10]. The *computation energy* is needed for performing arithmetic operations, especially the so-called multiply-and-accumulate (MAC) operations ($S \leftarrow S + wx$ on floats S, w, x), used to compute the weighted sums of inputs of the neurons. The *data energy* is required for moving the data inside the memory hierarchy of the hardware (dataflow), and is related to the number of memory accesses.

In a recent paper [7], we have introduced a simplified hardware-independent model of energy complexity for convolutional neural networks (CNNs). This model abstracts from the hardware implementation details related to different platforms, and preserves the asymptotic energy complexity of the CNN inference. It is composed of only two memory levels called *DRAM* and *Buffer*, illustrated in Fig. 1. The network parameters and states are stored in DRAM, and the arithmetic operations are performed over numerical data stored in Buffer, which is of a limited capacity of B bits. The transfer of data between the two memories is the dataflow. The main idea behind this model is that, for a given CNN stored in DRAM, the three arguments of all the MAC operations (i.e., input x, weight w and accumulated output S of operation $S \leftarrow S + wx$) employed for the evaluation of the network must occur together at the same time in Buffer. This process requires a certain number of data transfers between DRAM and Buffer (i.e., the number of DRAM accesses multiplied by the number of bits in a float number), which corresponds to our measure of the data energy.

For simplicity, we assume that the energy cost is not optimized across multiple CNN layers, as for instance in [1]. Hence, the energy complexity is defined as

a simple sum over separate convolutional and fully-connected layers only, while the less energy-intensive max pooling layers are omitted. Formally,

$$E = \sum_{\text{non-pooling layer } \lambda} \left(E^{\lambda}_{\text{comp}} + E^{\lambda}_{\text{data}} \right) \tag{1}$$

where the computation energy $E^{\lambda}_{\text{comp}}$ and the data energy $E^{\lambda}_{\text{data}}$ for evaluating a non-pooling layer λ is proportional to the corresponding numbers of MACs and DRAM accesses, respectively.

The energy complexity model of CNNs has been exploited for calculating the theoretical energy of processing convolutional layers in the context of two common dataflows and under realistic buffer capacity constraints [7]. For the first dataflow, any input to each neuron is read into Buffer only once. For the second one, any accumulated output of each neuron is written to DRAM only once. In both cases, each weight of the CNN is read into Buffer only once. These dataflows provide upper bounds on the energy complexity of CNNs, which have been compared to the real power consumptions estimated for Simba [6] and Eyeriss [2] architectures by using the Timeloop/Accelergy software tool [5, 9]. As it turns out, the theoretical upper bounds fit asymptotically very well the empirical optimal power consumptions, when individual parameters such as the height, width, depth, kernel size, and stride of a convolutional layer are varied [7]. Hence, the introduced energy complexity model appears to be capable of asymptotically capturing all important sources of energy consumption that are common to the diverse CNN hardware implementations.

The model can also be exploited for proving lower bounds on the energy complexity of CNNs, in order to establish asymptotic limits on the energy efficiency of any CNN hardware accelerators. Here, we start this study by investigating the case of *fully-connected layers*, as a specific case of convolutional layers. We first present two types of dataflows in which each weight and each output (or alternatively each input) are read into Buffer only once. In the first dataflow, the Buffer memory is assumed to be partitioned into two fixed parts of given capacities for inputs and outputs, respectively. The second dataflow is parameterized by the maximum number of inputs residing in Buffer at the same time. We determine the data energy complexity of both dataflows, which provides upper bounds for the optimal energy complexity. For the first dataflow, we further prove a matching lower bound by means of the weak duality theorem from linear programming. The optimal energy complexity for fully-connected layers in situations where Buffer is partitioned into two fixed parts ensues. The results are partially generalized to contiguous Buffer and are intended to be extended to convolutional layers in a future research.

The paper is organized as follows. Section 2 formally defines the energy complexity for fully-connected layers, and derives a general lower bound on the energy. Section 3 present two dataflows with their associated upper bounds on the energy. In Sect. 4, a matching and thus optimal lower bound is derived for the case of partitioned Buffer, and a partial generalization to contiguous Buffer is provided. Section 5 summarizes the results and discusses open problems.

2 Energy Complexity of Fully-Connected Layer

For simplicity, we consider a fully-connected CNN layer λ, which is composed of m neurons (units), each of which receiving connections labeled with real weights from all the n neurons in the previous layer $\lambda - 1$. This can be viewed as a complete weighted bipartite graph $G = (X, Y, E)$ where $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ are disjoint sets of $n = |X|$ *inputs* and $m = |Y|$ *outputs*, respectively, and $E = X \times Y$ is a set of directed edges (x_i, y_j) leading from input x_i to output y_j, each labeled with a real *weight* w_{ji}, for every $j = 1, \ldots, m$ and $i = 1, \ldots, n$. The fully-connected CNN layer is evaluated as follows:

$$y_j = \mathrm{ReLU}\left(w_{j0} + \sum_{i=1}^{n} w_{ji} x_i\right) \quad \text{for every } j = 1, \ldots, m\,, \tag{2}$$

where $\mathrm{ReLU}(x) = \max(0, x)$ is the rectified linear unit activation function and w_{j0} is a *bias* of output neuron y_j, for every $j = 1, \ldots, m$.

To evaluate the computation energy $E^{\lambda}_{\mathrm{comp}}$ of fully-connected layer λ in (1), note that the total number of MAC operations needed for computing (2) is mn: each output y_j is initialized with bias w_{j0} and requires n MAC updates to be computed. The computation energy is thus given by

$$E^{\lambda}_{\mathrm{comp}} = C_b\, mn \tag{3}$$

where C_b is a non-uniform parameter depending on the number of bits b in floating-point MAC operations, since the design of a MAC circuit inside a microprocessor differs for each b.

We now focus on the data energy $E^{\lambda}_{\mathrm{data}}$ of fully-connected layer λ in (1). This energy cost can be split into three components that count the DRAM accesses separately for the outputs, inputs, and weights:

$$E^{\lambda}_{\mathrm{data}} = E^{\lambda}_{\mathrm{outputs}} + E^{\lambda}_{\mathrm{inputs}} + E^{\lambda}_{\mathrm{weights}}\,. \tag{4}$$

In order to evaluate the sums in (2), all the mn couples of inputs and (accumulated) outputs (i.e. partially evaluated sums) need to occur in Buffer at least once. Each such pair (x_i, y_j) is associated with the unique weight w_{ji} that can be read from DRAM when the pair meet in Buffer for the first time. This means that each of the mn weights is read only once. Let ν and μ be the numbers of DRAM accesses to read inputs and outputs (or biases when initialized), respectively, and b be the number of bits in the floating point representation of outputs, inputs, and weights. The data energy (4) can thus be rewritten as

$$E^{\lambda}_{\mathrm{data}} = b\,(2\mu + \nu + mn) \tag{5}$$

since each output that is read into Buffer is later written back to DRAM, which corresponds to two DRAM accesses, whereas each input and weight are only read into Buffer. In order to optimize the data energy (4), it is thus sufficient to minimize $2\mu + \nu$.

We will now derive a simple general lower bound on the data energy (4) for fully-connected layers. Assume that Buffer has a size of $B = b(\beta + 1)$ bits, where $\beta > 1$ floats are reserved for storing inputs and outputs, and the remaining capacity of one float is dedicated to the weights. For notational simplicity, suppose that $\beta - 1$ divides m. In addition, for any dataflow, let r be the minimum number of phases during which either only inputs or only outputs are read into Buffer consecutively. Note that by reading a single input or output into Buffer, one can get at most $\beta - 1$ new input-output pairs in Buffer. Since all mn pairs need to meet in Buffer, we obtain the following trivial lower bound on the number of DRAM read accesses:

$$\mu + \nu \geq \frac{mn}{\beta - 1}\,. \tag{6}$$

Moreover, in order to keep generating new pairs in Buffer, at most n inputs or m outputs can be read during each phase. This ensures that $r(\beta - 1)\max(m, n) \geq mn$ which implies

$$r \geq \frac{\min(m, n)}{\beta - 1}\,. \tag{7}$$

Observe that, when a next phase begins, the reading of an input immediately after an output has been read (or vice versa) provides at most β new pairs in Buffer through these two DRAM read accesses (cf. the trivial upper bound $2(\beta - 1)$ of new pairs counted in (6) for two reads). Indeed, if there are k inputs $(1 \leq k \leq \beta - 1)$ and $\beta - k$ outputs in Buffer, the reading of an input yields at most $\beta - k$ new pairs, while the subsequent reading of an output generates at most k new pairs, which sums up to at most β new pairs in total. Let s be the number of readings that do not occur at the beginning of a new phase. The following lower bound on the number of DRAM read accesses ensues:

$$\mu + \nu \geq 2r + s + 1 \tag{8}$$

with

$$\beta r + (\beta - 1)s \geq mn \tag{9}$$

because all the mn pairs have to occur in Buffer, the two readings at the beginning of each of the r phases generate at most β new pairs, and each of the remaining s readings produces at most $\beta - 1$ new pairs, except for the very first DRAM read access providing no pair.

Inequality (9) can be rewritten as

$$(\beta - 1)(2r + s) \geq mn + (\beta - 2)r \tag{10}$$

which implies

$$\mu + \nu \geq \frac{mn}{\beta - 1} + \frac{\beta - 2}{\beta - 1}\,r + 1 \geq \frac{mn}{\beta - 1} + \frac{\beta - 2}{(\beta - 1)^2}\min(m, n) + 1 \tag{11}$$

according to (8) and (7). Since the biases of all m outputs must first be read into Buffer, we have $\mu \geq m$, and thus

$$2\mu + \nu \geq \frac{mn}{\beta - 1} + m + \frac{\beta - 2}{(\beta - 1)^2}\min(m, n) + 1\,. \tag{12}$$

This provides a general lower bound on the data energy of fully-connected layer λ:

$$E^{\lambda}_{\mathrm{data}} \geq b\left(mn + \frac{m(n-1)}{\beta-1} + \frac{\beta}{\beta-1}m + \frac{\beta-2}{(\beta-1)^2}\min(m,n) + 1\right) \tag{13}$$

according to (5).

3 Upper Bounds on Energy Complexity

Any correct dataflow for processing a fully-connected layer can be described by a sequence of p sets $B_0, B_1, \ldots, B_p \subseteq X \cup Y$, each of which being composed of vertices in G, that represent the successive contents of Buffer (excluding weights) after each DRAM access to read an input or output, in the course of evaluating the sums in (2). The sequence satisfies the following conditions:

1. $B_0 = \emptyset$
2. $|B_i| \leq \beta$ for every $i = 1, \ldots, p$
3. $|B_i \setminus B_{i-1}| = 1$ and $|B_{i-1} \setminus B_i| \leq 1$ for every $i = 1, \ldots, p$
4. $Y \subseteq \bigcup_{x \in B_i} B_i$ for every $x \in X$,

and its length p is the total number of DRAM read accesses,

$$p = \mu + \nu\,. \tag{14}$$

Condition 1 assumes empty Buffer at the beginning, and Condition 2 guarantees that its size is not exceeded. Condition 3 ensures that, by reading a single input or output into Buffer, at most one input or output is overwritten. Condition 4 ensures that all of the outputs meet every input in Buffer.

In the two following subsections, we present two dataflows for fixed and bounded number of inputs in Buffer, respectively, such that each output is read into Buffer only once (i.e., when initialized by a corresponding bias), which means that

$$\mu = m\,. \tag{15}$$

Clearly, the role of inputs and outputs can be reversed in these dataflows.

3.1 Fixed Number of Inputs in Buffer

For the first dataflow, we assume that Buffer is partitioned into two fixed parts for inputs and outputs, respectively, and contains one more float for reading the weights. One part is reserved for storing d inputs and the second one to store $\beta - d$ outputs, where d is a fixed parameter such that $1 \leq d \leq \beta - 1$. The dataflow can be described by the following sequence of sets $B_0, B_1, \ldots, B_p$ that meet Conditions 1–4, $|B_i \cap X| \leq d$, and $|B_i \cap Y| \leq \beta - d$ for every $i = 1, \ldots, p$:

$$\emptyset, \{x_1\}, \{x_1, x_2\}, \ldots, \{x_1, \ldots, x_d\}, \tag{16}$$

$$\{x_1, \ldots, x_d, y_1\}, \{x_1, \ldots, x_d, y_1, y_2\}, \ldots, \{x_1, \ldots, x_d, y_1, \ldots, y_{\beta-d}\}, \tag{17}$$

$$\{x_{d+1}, x_2 \ldots, x_d, y_1, \ldots, y_{\beta-d}\}, \{x_{d+1}, x_{d+2}, x_3, \ldots, x_d, y_1, \ldots, y_{\beta-d}\}, \ldots, \{x_{n-d+1}, \ldots, x_n, y_1, \ldots, y_{\beta-d}\}, \quad (18)$$

$$\{x_{n-d+1}, \ldots, x_n, y_{\beta-d+1}, y_2, \ldots, y_{\beta-d}\}, \{x_{n-d+1}, \ldots, x_n, y_{\beta-d+1}, y_{\beta-d+2}, y_3, \ldots, y_{\beta-d}\}, \ldots, \{x_{n-d+1}, \ldots, x_n, y_{\beta-d+1}, \ldots, y_{2(\beta-d)}\}, \quad (19)$$

$$\{x_{n-d+1}, \ldots, x_{n-1}, x_{n-d}, y_{\beta-d+1}, \ldots, y_{2(\beta-d)}\}, \{x_{n-d+1}, \ldots, x_{n-2}, x_{n-d-1}, x_{n-d}, y_{\beta-d+1}, \ldots, y_{2(\beta-d)}\}, \ldots, \{x_1, \ldots, x_d, y_{\beta-d+1}, \ldots, y_{2(\beta-d)}\}, \quad \ldots \quad (20)$$

After an initialization where the first d inputs are read into Buffer (16), the dataflow alternates between two phases of reading $\beta - d$ outputs (17) (or (19) etc.) and reading $n - d$ inputs (18) (or (20) etc.), respectively, while overwriting the outputs in Buffer by new outputs (cf. (19)) and the inputs in Buffer by new inputs (cf. (20)). Apart from d reads at initialization, $\frac{m}{\beta-d}$ changes from the first phase to the second one are performed before each of the m outputs has been read into Buffer once, which implies

$$p = d + \frac{m}{\beta - d}\left((\beta - d) + (n - d)\right) = \frac{m(n-d)}{\beta - d} + m + d\,. \quad (21)$$

Hence, this dataflow provides an upper bound on the data energy of fully-connected layer λ:

$$E^{\lambda}_{\text{data}} \leq b\left(mn + \frac{m(n-d)}{\beta - d} + 2m + d\right) \quad (22)$$

according to (5), (14), and (15). This upper bound takes the smallest value for $d = 1$, provided that $n \geq \beta$, since $n \geq \beta$ is equivalent to

$$\frac{m(n-1)}{\beta - 1} \leq \frac{m(n-d)}{\beta - d}\,.$$

Furthermore, an alternative upper bound to (22) is obtained when the roles of the inputs and outputs are reversed in the dataflow (16)–(20):

$$E^{\lambda}_{\text{data}} \leq b\left(mn + \frac{2n(m-(\beta-d))}{d} + n + 2(\beta - d)\right)\,. \quad (23)$$

This upper bound has the smallest value for $d = \beta - 1$, provided that $m \geq \beta$, since $m \geq \beta$ is equivalent to

$$\frac{2n(m-1)}{\beta - 1} \leq \frac{2n(m-(\beta-d))}{d}\,.$$

Finally, assuming $n \geq \beta$ and $m \geq \beta$, we can compare (22) and (23) for their smallest values, namely $d = 1$ and $d = \beta - 1$, respectively:

$$b\left(mn + \frac{m(n-1)}{\beta - 1} + 2m + 1\right) \overset{?}{\leq} b\left(mn + \frac{2n(m-1)}{\beta - 1} + n + 2\right) \quad (24)$$

which can be rewritten as

$$0 \overset{?}{\leq} m(n-2\beta+3)+n(\beta-3)+\beta-1\,. \tag{25}$$

This inequality holds for $n > 2\beta-3$ implying $n \geq \beta$ due to $\beta \geq 2$. Therefore, we can conclude that for sufficiently large $n > 2\beta-3$ and $m \geq \beta$, the minimal energy for fully-connected layers achieved by the dataflow (16)–(20) is obtained when $d = 1$, i.e., when Buffer is partitioned to $\beta-1$ outputs, one input, and one weight. This situation leads to the following upper bound:

$$E^{\lambda}_{\text{data}} \leq b\left(mn+\frac{m(n-1)}{\beta-1}+2m+1\right)\,. \tag{26}$$

3.2 Bounded Number of Inputs in Buffer

The second dataflow is parameterized by the maximum number k of inputs that can simultaneously occur in Buffer, where $1 \leq k \leq \beta-1$. The dataflow is described by the following sequence of sets $B_0, B_1, \ldots, B_p$ satisfying Conditions 1–4 and $|B_i \cap X| \leq k$ for every $i = 1, \ldots, p$:

$$\emptyset, \{x_1\}, \{x_1, x_2\}, \ldots, \{x_1, \ldots, x_k\}, \tag{27}$$

$$\{x_1, \ldots, x_k, y_1\}, \{x_1, \ldots, x_k, y_1, y_2\}, \ldots, \{x_1, \ldots, x_k, y_1, \ldots, y_{\beta-k}\}, \tag{28}$$

$$\{x_1, \ldots, x_{k-1}, y_1, \ldots, y_{\beta-k+1}\}, \{x_1, \ldots, x_{k-2}, y_1, \ldots, y_{\beta-k+2}\}, \ldots,$$
$$\{x_1, y_1, \ldots, y_{\beta-1}\}, \tag{29}$$

$$\{x_n, y_1, \ldots, y_{\beta-1}\}, \{x_{n-1}, y_1, \ldots, y_{\beta-1}\}, \ldots, \{x_{k+1}, y_1, \ldots, y_{\beta-1}\}, \tag{30}$$

$$\{x_k, x_{k+1}, y_2, \ldots, y_{\beta-1}\}, \{x_{k-1}, x_k, x_{k+1}, y_3, \ldots, y_{\beta-1}\}, \ldots,$$
$$\{x_2, \ldots, x_{k+1}, y_k, \ldots, y_{\beta-1}\}, \tag{31}$$

$$\{x_2, \ldots, x_{k+1}, y_{k+1}, \ldots, y_{\beta}\}, \{x_2, \ldots, x_{k+1}, y_{k+2}, \ldots, y_{\beta+1}\}, \ldots,$$
$$\{x_2, \ldots, x_{k+1}, y_{\beta}, \ldots, y_{2\beta-k-1}\}, \tag{32}$$

$$\{x_2, \ldots, x_k, y_{\beta}, \ldots, y_{2\beta-k}\}, \{x_2, \ldots, x_{k-1}, y_{\beta}, \ldots, y_{2\beta-k+1}\}, \ldots,$$
$$\{x_2, y_{\beta}, \ldots, y_{2\beta-2}\}, \tag{33}$$

$$\{x_1, y_{\beta}, \ldots, y_{2\beta-2}\}, \{x_n, y_{\beta}, \ldots, y_{2\beta-2}\}, \{x_{n-1}, y_{\beta}, \ldots, y_{2\beta-2}\}, \ldots,$$
$$\{x_{k+2}, y_{\beta}, \ldots, y_{2\beta-2}\}, \tag{34}$$

$$\{x_{k+1}, x_{k+2}, y_{\beta+1}, \ldots, y_{2\beta-2}\}, \{x_k, x_{k+1}, x_{k+2}, y_{\beta+2}, \ldots, y_{2\beta-2}\}, \ldots,$$
$$\{x_3, \ldots, x_{k+2}, y_{\beta+k-1}, \ldots, y_{2\beta-2}\}, \quad \ldots \tag{35}$$

After an initialization when the first k inputs are read into Buffer (27), the dataflow alternates between two phases of reading $\beta-1$ outputs (28)–(29) (or (32)–(33) etc.) and reading $n-1$ inputs (30)–(31) (or (34)–(35) etc.), respectively. In the general first phase (32)–(33) (when outputs are read into Buffer), $\beta-k$ outputs currently stored in Buffer are first replaced by new ones (32), and only then the $k-1$ inputs residing in Buffer are overwritten by outputs (33) until one input remains in Buffer. During the second phase (34)–(35) (when inputs

are read into Buffer), the remaining input is being replaced one by one with $n-k$ inputs (34), and then the last $k-1$ read inputs overwrites the outputs stored in Buffer, so that k inputs and $\beta-k$ outputs are left in Buffer at the end of the second phase. This phase can again be followed by the first phase, etc. Apart from k reads at initialization, the first phase changes to the second one $\frac{m}{\beta-1}$ times before each of the m outputs is read into Buffer once, which implies

$$p = k + \frac{m}{\beta-1}\left((\beta-k)+(k-1)+(n-k)+(k-1)\right) = \frac{m(n-1)}{\beta-1}+m+k\,. \quad (36)$$

Hence, this dataflow provides an upper bound on the data energy of fully-connected layer λ:

$$E^{\lambda}_{\mathrm{data}} \le b\left(mn + \frac{m(n-1)}{\beta-1} + 2m + k\right) \quad (37)$$

according to (5), (14), and (15). Note that the first dataflow (16)–(20) for $d=1$ coincides with the second dataflow (27)–(35) for $k=1$, producing the same upper bound (26).

4 Lower Bounds on Energy Complexity

4.1 Partitioned Buffer

We now study the case where Buffer is divided into two fixed parts dedicated to the reading of d inputs and $\beta-d$ outputs, respectively, plus one float for weights, where d is a fixed parameter such that $1 \le d \le \beta-1$. In this context, we improve the general lower bound (13) on the data energy $E^{\lambda}_{\mathrm{data}}$ of fully-connected layer λ so that it matches the upper bounds (22) and (23), up to an additive constant. We distinguish two cases according to whether d is at most or at least $\frac{2}{3}\beta$.

Case $1 \le d \le \frac{2}{3}\beta$. Assume first that

$$1 \le d \le \tfrac{2}{3}\beta\,. \quad (38)$$

We formulate a linear program of finding μ and ν that

$$\begin{aligned} \text{minimize} \quad & 2\mu+\nu && (39)\\ \text{subject to} \quad & d\mu+(\beta-d)\nu \ge mn && (40)\\ & \mu \ge m && (41)\\ & \nu \ge 0\,, \quad \mu \ge 0\,. && (42) \end{aligned}$$

Constraint (40) expresses the fact that all mn input-output couples have to occur in Buffer, since by reading one output or input, at most d or $\beta-d$ new pairs meet in Buffer, respectively. Constraint (41) ensures that at least m outputs are read into Buffer. We convert the linear program (39)–(42) to the corresponding dual linear program of finding ϕ and ψ that

$$\begin{aligned} \text{maximize} \quad & mn\,\phi + m\,\psi && (43)\\ \text{subject to} \quad & d\phi+\psi \le 2 && (44) \end{aligned}$$

$$(\beta - d)\phi \leq 1 \tag{45}$$

$$\phi \geq 0, \quad \psi \geq 0\,. \tag{46}$$

Observe that $\phi_0 = \frac{1}{\beta - d}$ and $\psi_0 = 2 - \frac{d}{\beta - d}$ is a feasible solution for the dual program, satisfying (44)–(46) due to (38). By the weak duality theorem, the objective function value of the primal (39) at any feasible solution is lower bounded by the objective function value of the dual (43) at any feasible solution, that is,

$$2\mu + \nu \geq mn\,\phi_0 + m\,\psi_0 = \frac{m(n-d)}{\beta - d} + 2m\,. \tag{47}$$

According to (5), inequality (47) provides the following lower bound on the data complexity of fully-connected layer λ:

$$E^{\lambda}_{\text{data}} \geq b\left(mn + \frac{m(n-d)}{\beta - d} + 2m\right) \tag{48}$$

when Buffer is divided into two parts for d inputs and $\beta - d$ outputs, and the fixed parameter d meets (38). This lower bound matches the corresponding upper bound (22) achieved by the dataflow (16)–(20), up to the additive constant d.

Case $\frac{2}{3}\beta \leq d \leq \beta - 1$. Similarly, for

$$\tfrac{2}{3}\beta \leq d \leq \beta - 1\,, \tag{49}$$

we have a linear program of finding μ and ν that minimize $2\mu + \nu$ subject to $d\mu + (\beta - d)\nu \geq mn$, $\nu \geq n$, $\nu \geq 0$, and $\mu \geq 0$. This is converted to the corresponding dual linear program of finding ϕ and ψ that maximize $mn\,\phi + n\,\psi$ subject to $d\phi \leq 2$, $(\beta - d)\phi + \psi \leq 1$, $\psi \geq 0$, and $\psi \geq 0$, which has a feasible solution $\phi_1 = \frac{2}{d}$ and $\psi_1 = 1 - \frac{2(\beta - d)}{d}$ due to (49). By the weak duality theorem we have

$$2\mu + \nu \geq mn\,\phi_1 + n\,\psi_1 = \frac{2n(m - (\beta - d))}{d} + n \tag{50}$$

which provides the following lower bound on the data complexity of fully-connected layer λ:

$$E^{\lambda}_{\text{data}} \geq b\left(mn + \frac{2n(m - (\beta - d))}{d} + n\right) \tag{51}$$

when Buffer is divided into two parts for d inputs and $\beta - d$ outputs, and the fixed parameter d meets (49). This lower bound matches the corresponding upper bound (23) achieved by the dataflow (16)–(20) with the reversed role of inputs and outputs, up to the additive constant $2(\beta - d)$.

We can conclude that the data energy for fully-connected layers achieved by the dataflow (16)–(20) when Buffer is partitioned to d inputs, $\beta - d$ outputs, and one weight, is optimal for any fixed d, and the minimum of data energy (26) is achieved for $d = 1$.

4.2 Partial Generalization

In general case when Buffer is not divided into separate parts, the lower bound (13) on the data energy complexity still differs from the upper bound (26) by linear additive term $\frac{\beta-2}{\beta-1}\big(m-\frac{\min(m,n)}{\beta-1}\big)$, which can further be improved in some special cases. In particular, denote by μ_k and ν_k for $1 \le k \le \beta-1$ the number of accesses to DRAM for reading outputs and inputs at the points when exactly k inputs reside in Buffer. The linear program (39)–(42) can be generalized to the following program of finding μ_k and ν_k for $1 \le k \le \beta-1$ that

$$\text{minimize} \quad 2\mu+\nu = 2\sum_{k=1}^{\beta-1}\mu_k + \sum_{k=1}^{\beta-1}\nu_k \tag{52}$$

$$\text{subject to} \quad \sum_{k=1}^{\beta-1} k\mu_k + \sum_{k=1}^{\beta-1}(\beta-k)\nu_k \ge mn \tag{53}$$

$$\sum_{k=1}^{\beta-1}\mu_k \ge m \tag{54}$$

$$\mu_k \ge 0\,, \quad \nu_k \ge 0 \quad \text{for } k=1,\dots,\beta-1\,. \tag{55}$$

By applying the weak duality theorem to this program, one can achieve only the trivial lower bound (6). Nevertheless, this lower bound can be improved when the following, yet somewhat artificial, condition is added to (53)–(55):

$$\nu_k - \mu_k \ge 0 \quad \text{for } k=1,\dots,\beta-1\,, \tag{56}$$

that is, $\nu_k \ge \mu_k$ for $1 \le k \le \beta-1$. This condition states that input readings into Buffer is preferred over more expensive output readings, since outputs need to be written back to DRAM. Note that this condition is satisfied by the dataflows presented in Sect. 3.

Thus, we convert the linear program (52)–(56) to the corresponding dual linear program of finding ϕ, ψ, and χ_k for $1 \le k \le \beta-1$, that maximize $mn\,\phi + m\,\psi$ subject to $k\phi+\psi-\chi_k \le 2$, $(\beta-k)\phi+\chi_k \le 1$, $\phi \ge 0$, $\psi \ge 0$, and $\chi_k \ge 0$ for every $k=1,\dots,\beta-1$. Observe that $\phi_0 = \frac{1}{\beta-1}$, $\psi_0 = 2-\frac{1}{\beta-1}$, and $\chi_{k0} = \frac{k-1}{\beta-1}$ for $1 \le k \le \beta-1$, is a feasible solution for this dual. By the weak duality theorem, we have

$$2\mu+\nu = 2\sum_{k=1}^{\beta-1}\mu_k + \sum_{k=1}^{\beta-1}\nu_k \ge mn\,\phi_0 + m\,\psi_0 = \frac{m(n-1)}{\beta-1} + 2m \tag{57}$$

which proves the optimality of the data energy (26) (up to 1) also for contiguous Buffer, provided that condition (56) holds.

5 Conclusion

In this paper, we have theoretically analyzed the energy complexity model for CNNs introduced in our previous work [7], which is asymptotically consistent with estimates of power consumption for different CNN hardware implementations. We have confined ourselves to fully-connected layers as a starting point for the future analysis of convolutional layers. We have shown a simple general lower bound on energy complexity of fully-connected layers. We have presented two dataflows for fixed and bounded numbers of inputs residing in Buffer, respectively, and computed their energy costs to obtain upper bounds on the energy. We have then proven a matching lower bound on the energy for the first dataflow, which in turn provides the optimal energy complexity for fully-connected layers when Buffer is partitioned into two fixed parts for inputs and outputs.

In future research, the lower bound is intended to be generalized to contiguous Buffer, namely Buffer without partition. The partial generalization presented here shows that a linear program formulation seems to be not strong enough to achieve this goal, meaning that a detailed analysis of DRAM accesses would be needed. This analysis could then be used to prove the optimal energy complexity for convolutional layers, which represents the main challenge of this research.

Acknowledgements. The research was partially supported by the institutional support RVO: 67985807 and the Czech Science Foundation grant GA22-02067S. We thank Petr Savický for inspiring discussions in the early stages of this research.

References

1. Alwani, M., Chen, H., Ferdman, M., Milder, P.A.: Fused-layer CNN accelerators. In: Proceedings of IEEE/ACM MICRO 2016, pp. 22:1–22:12 (2016). https://doi.org/10.1109/MICRO.2016.7783725
2. Chen, Y., Emer, J.S., Sze, V.: Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. In: Proceedings of ACM/IEEE ISCA 2016, pp. 367–379 (2016). https://doi.org/10.1109/ISCA.2016.40
3. Mittal, S.: A survey of techniques for approximate computing. ACM Comput. Surv. **48**(4), 62:1–62:33 (2016). https://doi.org/10.1145/2893356
4. Mittal, S.: A survey of FPGA-based accelerators for convolutional neural networks. Neural Comput. Appl. **32**(4), 1109–1139 (2020). https://doi.org/10.1007/s00521-018-3761-1
5. Parashar, A., et al.: Timeloop: A systematic approach to DNN accelerator evaluation. In: Proceedings of IEEE ISPASS 2019, pp. 304–315 (2019). https://doi.org/10.1109/ISPASS.2019.00042
6. Shao, Y.S., et al.: Simba: Scaling deep-learning inference with multi-chip-module-based architecture. In: Proceedings of IEEE/ACM MICRO 2019, pp. 14–27 (2019). https://doi.org/10.1145/3352460.3358302
7. Šíma, J., Vidnerová, P., Mrázek, V.: Energy complexity model for convolutional neural networks. In: Proceedings of ICANN 2023. LNCS, Springer (2023)
8. Sze, V., Chen, Y., Yang, T., Emer, J.S.: Efficient Processing of Deep Neural Networks. Synthesis Lectures on Computer Architecture, Morgan & Claypool Publishers (2020). https://doi.org/10.2200/S01004ED1V01Y202004CAC050

9. Wu, Y.N., Emer, J.S., Sze, V.: Accelergy: An architecture-level energy estimation methodology for accelerator designs. In: Proceedings of IEEE/ACM ICCAD 2019 (2019). https://doi.org/10.1109/ICCAD45719.2019.8942149
10. Yang, T., Chen, Y., Emer, J.S., Sze, V.: A method to estimate the energy consumption of deep neural networks. In: Proceedings of IEEE ACSSC 2017, pp. 1916–1920 (2017). https://doi.org/10.1109/ACSSC.2017.8335698