2017 IEEE 35th International Conference on Computer Design

Applications of Deep Neural Networks for Ultra Low Power IoT

(Invited Paper)

Sreela Kodali[2,*], Patrick Hansen[1,*], Niamh Mulholland[1], Paul Whatmough[1,3], David Brooks[1], and Gu-Yeon Wei[1]

[1]Harvard University, Cambridge, MA
[2]Princeton University, Princeton, NJ
[3]ARM Research, Boston, MA
[*]Equal contributions

***Abstract*—IoT devices are increasing in prevalence and popularity, becoming an indispensable part of daily life. Despite the stringent energy and computational constraints of IoT systems, specialized hardware can enable energy-efficient sensor-data classification in an increasingly diverse range of IoT applications. This paper demonstrates seven different IoT applications using a fully-connected deep neural network (FC-NN) accelerator on 28nm CMOS. The applications include audio keyword spotting, face recognition, and human activity recognition. For each application, a FC-NN model was trained from a preprocessed dataset and mapped to the accelerator. Experimental results indicate the models retained their state-of-the-art accuracy on the accelerator across a broad range of frequencies and voltages. Real-time energy results for the applications were found to be on the order of 100nJ per inference or lower.**

I. Introduction

From fitness and health to security and home appliances, IoT devices are ubiquitous and have become an indispensable part of daily life. Introducing advanced sensing and intelligence to a plethora of domains, IoT has enabled groundbreaking applications [1]. Many IoT systems require continuous sensing and sensor-data classification, but these tasks are exacting and intense for the system 's finite energy and computational resources. Traditional microcontrollers consume a lot of power to perform the appropriate computations. However, specialized hardware can bypass these costly calculations and allow for energy-efficient classification in IoT devices. Doing so facilitates more sophisticated, diverse, and meaningful IoT workloads. This paper explores seven different IoT workloads spanning audio keyword spotting, face recognition, and human activity recognition (HAR) applications. These workloads all employ neural networks. Each of the workloads utilizes a dataset that, after preprocessing, is used to train a fully-connected neural network (FC-NN). These FC-NN models were ported onto a 28nm FC-NN hardware accelerator SoC [2]. Experimental results demonstrate high classification performance and low power consumption for each of the workloads, with real-time energy costs on the order of nJs per inference.

II. IoT Applications

After an extensive literature review, deep learning datasets were selected to create seven distinct IoT related applications. The datasets encompass a wide range of applications including audio keyword spotting, face recognition, and HAR.

A. Overview of Datasets

DARPA Resource Management (RM2) consists of digital and transcribed speech for the use in training and evaluating speech recognition and keyword spotting systems, recorded at 16KHz with 16-bit resolution [3]. The subset of RM2 used to train and test our models contains 5,450 utterances from 168 speakers who collectively represent a wide variety of American dialects. The complete lexicon of RM2 contains more than 1,000 words with average word length of 300ms, and each model evaluates detection of a subset of these words. Feature extraction is performed similarly to prior literature [4]. For every 10ms of speech, the first 13 MFCCs[1] are extracted from the raw audio waveforms within a 25ms window. Each input vector contains all extracted MFCCs within a 31 frame window, resulting in 403 features per frame. These input vectors are normalized to have zero mean and unit variance. A moving average window is swept across the outputs of the FC-NN. Thresholds that maximize the F-score for each keyword are chosen from predictions on the training set, and those thresholds are used for inference on the test set.

Labeled Faces in the Wild (LFW) was designed for studying the problem of unconstrained face recognition [5]. LFW contains over 13,000 labeled images of faces of more than 5,700 individuals. The images in this dataset contain a high degree of variation in pose, lighting, facial expression, age, gender, race, and background. Ths variation is motivated by replicating a natural degree of variation in images that could be seen by real-world facial recognition applications. The 250x250 pixel color images are aligned by funneling [6], cropped to 128x128, and converted to grayscale. The matrix of all training images is used to fit a projection matrix using principal component analysis (PCA), which is then

[1]Mel-frequency cepstral coefficients

used to project image vectors onto a lower-dimensional space. Each input vector to the FC-NN is the concatenation of two PCA-reduced image vectors, representing a pair of images in which the faces either match or do not match. LFW defines several paradigms for reporting performance. Due to the use of funneling for alignment, the performance reported in this paper is of the *Image-Restricted, Label-Free Outside Data* paradigm.

OPPORTUNITY Activity Recognition (OPP) dataset consists of wearable data from subjects in a simulated breakfast scenario [7]. Focusing on the activities of daily living, this dataset serves as a benchmark for human activity recognition algorithms. Seven IMUs, shown in figure 1a, recorded data at 30 Hz of 18 unique gestures (i.e. open drawer, open fridge). Following suit of existing literature [8], the subset of sensory data with no packet loss was used to train and develop the FC-NN model. Each sample size had 77 features, and a total of 650K samples were available. Run 2 of subject 1 was the validation data, runs 4 and 5 of subjects 2 and 3 was used as the test data, and the rest of the runs served as the training data. [8] Time-series segmentation, a preprocessing technique to contextualize time-dependent information, was applied to the raw data with a window size of 0.36 seconds (12 samples) and 50% overlap.

PAMAP2 Physical Activity Monitoring dataset includes data from subjects performing 13 different physical activities, from walking and cycling to vacuum cleaning and ironing [9]. PAMAP2 captures a broad range of conventional physical activities. Subjects wore 3 IMUs, indicated in figure 1b, collecting data at 100Hz and a heart rate monitor collecting data at 9Hz. Samples had 52 dimensions and the dataset comprised of 473K samples. Replicating processing from existing literature (hammerla paper), the raw data was downsampled to have a similar temporal resolution as the OPPORTUNITY dataset. Additionally, the values were rescaled to fit within 5-bit values, and missing value handling was resolved with linear interpolation. Only data from the "protocol" subset was used. Runs 1 and 2 of subject 5 were the validation data, runs 1 and 2 of subject 6 were the test data, and the remaining runs served as the training data [8]. Time-series segmentation was applied to the raw data with a window size of 0.56 seconds (18 samples) and -10% overlap. Negative overlap fraction indicates a space between the sampling windows.

Daphnet Freezing of Gait (DG) dataset comprises acceleration sensor data from patients with Parkinson 's Disease, a progressive disorder impacting the motor system [10]. The third human activity dataset, Daphnet Freezing of Gait, benchmarks algorithms for identifying walking gait freeze and aims to address a pressing health concern. Subjects wore 3 sensors on their hips and legs as in figure 1c and performed activities of daily living. The data was collected at 64Hz and was annotated by professionals as either "freezing", "not freezing", or "not part of the experiment." Each data sample had 9 features, and 470K samples were available. Similar to PAMAP2, the raw data was downsampled, rescaled, and missing value handling was accomplished with linear interpolation. Run 1 of subject 9 was the validation data, runs 1 and 2 of subject 2 were the test data, and the remaining data was used for training [8]. Time-series segmentation was applied to the raw data with a window size of 0.48 seconds (16 samples) and 50% overlap.

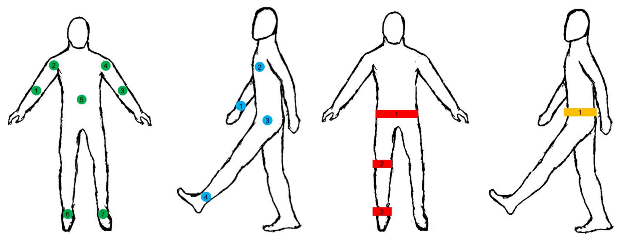
(a) OPP (b) PAMAP2 (c) DG (d) Smartphone

Fig. 1. Sensors for HAR datasets: (a) sensors 1-7 are IMUs positioned on arms, back, and ankles; (b) sensor 1 is a wrist heart-rate monitor, and sensors 2-4 are IMUs positioned on the chest, hip, and ankle; (c) sensor 1-3 are accelerometers on the trunk, thigh, and ankle; (d) sensor 1 is a smartphone at hip-level.

Smartphone-based Human Activity Recognition: Raw subset is one of two subsets of the Smartphone-based HAR dataset. This subset includes the raw accelerometer and gyroscope data at 64Hz from a Samsung Galaxy S2 smartphone fastened at the waist of subjects (as in figure 1d) doing 13 simple static and dynamic activities [11]. The dataset sets a standard for basic activity recognition with minimal sensing hardware. Each sample has 6 dimensions: the x, y, and z components of acceleration and of angular speed. Experiments 44 to 50 were validation data, experiments 51 to 61 were test data, and the remaining experiments were training data. Time-series segmentation was applied to the data with a window size of 2.56 seconds (128 samples) and 50% overlap.

Smartphone-based Human Activity Recognition: Feature-extracted (FE) subset is the second subset of the Smartphone-based HAR dataset. The feature-extracted subset comprises very specific values derived from the raw smartphone data (e.g., body acceleration, gravity acceleration, mean, standard deviation) [11]. Each sample has 561 dimensions, and the FE subset is preprocessed and partitioned as provided [11].

B. *FC-NN Model Design*

Keras, a Python deep learning framework with Theano backend, was used to build the FC-NN models for the datasets. Each model consists of an input layer, 2-4 hidden layers, and an output layer. The exact topology was selected to maximize the performance metrics, rather than minimizing operations or model size.

For training, Adadelta was used with the default Keras parameter values of: $lr = 1.0, \rho = 0.95, \epsilon = 1 \times 10^{-8}$. No learning rate decay was used. We used a batch size of 256 examples and trained the models for as many epochs as it took for the accuracy to converge. Other hyperparameters were chosen by performing a grid search. Table I summarizes the models.

TABLE I
APPLICATION DATASETS, MODELS, AND EVALUATED PERFORMANCE

Dataset			Model		Performance		
Name	Inputs	Outputs	Topology	Size (MB)	Acc (%)	Mean F-score	Weighted F-score
RM2	403	10	200-200	0.235	99.1	0.837	0.983
LFW	512	1	128-128-128	0.188	78.4	0.773	0.773
OPP	924	18	240-240-240-240	0.800	90.8	0.629	0.904
PAMAP2	936	13	220-220-220-220	0.710	72.2	0.665	0.716
DG	144	3	112-112-112-112	0.104	71.9	0.617	0.711
Smartphone (Raw)	384	13	256-256-256-256	0.599	80.3	0.703	0.800
Smartphone (FE)	561	12	280-280-280-280	0.794	93.6	0.817	0.935

C. Model Performance

Model performance was evaluated by test set accuracy, mean F-score, and the weighted F-score (Table I). Since IoT datasets are often heavily skewed[2], test accuracy alone is not a robust metric for performance. Instead, F-score, which is the harmonic mean of precision[3] and recall[4], considers accuracy for every class and is more suitable for skewed datasets. Mean f-score F_m and weighted F-score F_w are defined as

$$F_m = \frac{2}{|c|}\sum_c \frac{precision_c \times recall_c}{precision_c + recall_c}$$

$$F_w = 2\sum_c \frac{N_c}{N_T}\frac{precision_c \times recall_c}{precision_c + recall_c}$$

where N_c is the number of samples in class c, and N_T is the total number of samples [8].

III. EXPERIMENTAL RESULTS

A. Experimental Setup

In this section, we present measured results of the seven IoT applications running on a 28nm FC-NN hardware accelerator test chip [2]. Optimal frequency-voltage requirements and power consumption of the datasets are determined in each case. For frequencies ranging from 200MHz - 1.3GHz, the hardware accelerator can operate with a supply voltage as low as 0.56V. Figure 2 shows the minimum clock frequency required for a given accelerator voltage when evaluated on a model for the MNIST dataset evaluated in [2]. From this data, we can extrapolate the minimum power consumption for each of our models.

B. Mapping to Hardware Accelerator

The hardware accelerator has a number of constraints that must be met to execute the models. Firstly, there is only 1MB of memory available on the accelerator for storing the model parameters. Hence, the weights and biases must fit within that space. These weights can be represented as either 8-bit or 16-bit fixed-precision numbers. In the case of 16-bit, this allows for no more than 500K parameters. Several of the models (RM2, LFW, and DG) are far smaller than the allowed 1MB; in

[2]Most samples do not belong to a specific class and are labeled as "Other".

[3]The percentage of samples belonging to a class out of all samples that have been predicted to belong to that class.

[4]The percentage of samples correctly predicted to belong to a class out of all samples that truly belong to that class.

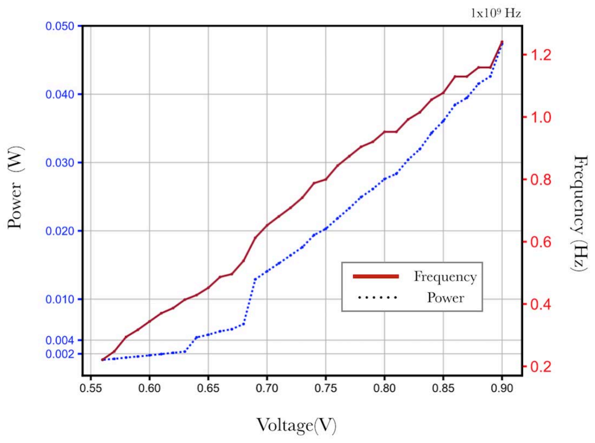

Fig. 2. Measured power and frequency of 28nm hardware accelerator test chip [2] over a range of supply voltages. Each operating point corresponds to the lowest error-free voltage at a given frequency.

these cases, increasing the number of free parameters did not increase accuracy. This is likely due to the data preprocessing preventing generalization of the data and the models became prone to overfitting with larger capacity models.

Secondly, the number of elements in the activation vector for each layer cannot exceed 1K. Therefore the input feature lengths or hidden layer sizes must fit this constraint. This influenced decisions regarding feature reduction techniques applied to inputs of the models.

The first step of mapping the models into hardware is to extract the weights and biases from the Keras model, and to translate them from 32-bit floating point values to 16-bit fixed point values which are used for computation by the accelerator, which is done simply by quantizing the parameters and storing them in the required format. The test vectors must also be quantized. The weights are represented as fixed-point numbers with 2 integer bits and 14 fractional bits, and the input features as fixed-point numbers with 4 integer bits and 12 fractional bits. This reduction in precision from 32-bit to 16-bit does contribute some small amount of error during inference. However this contribution is found to be insignificant, and so we did not retrain the models after they had been quantized.

When evaluated on the hardware accelerator, the accuracies and weighted F-scores of all the models varied by less than 1%

TABLE II
APPLICATION POWER REQUIREMENTS

Dataset	Inference Rate (R_I)	Clock Cycles	Freq (kHz)	Power (nW)	Energy per Inference (nJ)
RM2	100.00	16,138	1,613.8	4,070.4	40.7
LFW	10.00	13,075	130.8	330.4	33.0
OPP	5.56	51,368	285.6	722.0	129.9
PAMAP2	1.62	51,420	83.3	210.6	130.0
DG	4.17	7,092	29.6	74.8	17.9
Smartphone (Raw)	1.56	37,979	59.2	149.7	96.0
Smartphone (FE)	0.78	50,662	39.5	99.9	128.1

in comparison to the floating point versions. The mean F-score for the OPPORTUNITY model dropped by slightly more than 1%, due to a change in accuracy in a highly underrepresented class. Because the weighted F-score considers the number of test examples in each class, the loss in accuracy of that single class proved to have an insignificant impact on the overall accuracy of the model.

C. Real-time Energy

To calculate the minimum power consumption required by a real-time IoT application utilizing one of these models, we first determine the minimum clock frequency required for each of the workloads. Then from that frequency we determine the minimum supply voltage required for correct operation of the accelerator without introducing faults into the system that could compromise inference accuracy. Once we have determined the supply voltage and clock frequency, we can then determine the expected power consumption for each workload.

The minimum clock frequency can be determined simply by considering two attributes of the model: the number of inferences required per unit time by the application, and the number of clock cycles per inference a model takes to complete on the accelerator. For LFW, we estimate the number of required inferences per second to be 10 for standard facial recognition applications that could be used to unlock mobile devices. The other models use sliding, overlapping windows to produce input features, and the real-time constraint for these models is

$$R_I = \frac{1}{w(1-o)}$$

where the inference rate R_I is determined by the dataset preprocessing window size w and overlap fraction o.

The number of cycles per inference is determined by using on-chip performance counters, which report various statistics, including the total number of clock cycles consumed per inference. The clock frequency required for all the applications is lower than the critical frequency at the minimum supply voltage V_{min}, and so we assume V_{min} for power measurement of all workloads. This operating point has a clock frequency of 443MHz and a power consumption of 1.12mW. Using this optimal operating point, we derived the estimated power consumption for each of the models given their real-time constraints, which are provided in Table II.

IV. Conclusion

This paper presents seven distinct deep learning IoT applications. On these workloads we have demonstrated high prediction accuracy while also achieving ultra low energy consumption utilizing a FC-NN accelerator. The energy consumption per inference for each of our models is on the order of 100nJ or lower. These results demonstrate the benefits of specialized hardware and their potential to contribute in the IoT domain. Future work may include exploring different datasets, evaluating FC-NN models with different latencies, and deploying an end-to-end IoT system.

Acknowledgements

This work was supported in part by DARPA (HR0011-13-C-0022) and Intel Corporation. This research was, in part, funded by the U.S. Government. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

References

[1] L. D. Xu, et al., *Internet of Things in Industries: A Survey*, IEEE Trans. on Industrial Informatics, 2014.

[2] P. N. Whatmough, et al., *A 28nm SoC with a 1.2GHz 568nJ/prediction sparse deep-neural-network engine with 0.1 timing error rate tolerance for IoT applications*, IEEE Int. Solid-State Circuits Conference (ISSCC), 2017.

[3] P. Price, et al. Resource Management RM2 2.0 LDC93S3C. DVD. Philadelphia: Linguistic Data Consortium, 1993.

[4] M. Shah, et al., *A fixed-point neural network for keyword detection on resource constrained hardware*, IEEE Workshop on Signal Processing Systems, 2015.

[5] Gary B. Huang, et al., *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*, University of Massachusetts, Amherst, Technical Report, 2007.

[6] Gary B. Huang, et al., *Unsupervised joint alignment of complex images*, Int. Conf. on Computer Vision (ICCV), 2007.

[7] Ricardo Chavarriaga, et al., *The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition*, Pattern Recognition Letters, 2013.

[8] Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. *Deep, convolutional, and recurrent models for human activity recognition using wearables*, arXiv:1604.08880v1.

[9] Reiss and D. Stricker, *Introducing a New Benchmarked Dataset for Activity Monitoring*, Int. Symp. on Wearable Computers (ISWC), 2012.

[10] Marc Bchlin, et al., *Wearable Assistant for Parkinson's Disease Patients With the Freezing of Gait Symptom*, IEEE Trans. on Information Technology in Biomedicine, 2010.

[11] Jorge-L. Reyes-Ortiz, et al., *Transition-Aware Human Activity Recognition Using Smartphones*, Neurocomputing, Springer, 2015.