Sensor Applications

Predictive Sampling in Image Sensing for Sparse Image Processing

Amin Biglari[1], Qisong Hu[1], and Wei Tang[1*]

[1]*Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM, 88001, USA*
[*]*Member, IEEE*

Manuscript received September. 7, 2024; revised September 21, 2024; accepted October 6, 2024. Date of publication October 12, 2017; date of current version October 12, 2024.

Abstract—This paper presents a pixel-level predictive sampling method for image sensing and processing to reduce the computing overhead for power-limited image sensing systems. The predictive sampling method scans through rows and columns to identify the location and value of the critical pixels, which are the turning points in the row and column arrays. The prediction is performed using the value of prior pixels and a pre-defined error threshold. When the prediction is successful, the pixel is marked as a non-critical pixel and is skipped for recording and processing. Only the critical pixels are selected for further processing. We proposed reconstruction methods that recover the raw image from the selected critical pixels using interpolation. The experimental results show that the proposed method can reduce the data throughput by 72% with an error of 1.6% for sparse images. The convolutional neural network model applied with this method can achieve a similar detection accuracy in a standard method while only using 27.1% of data size.

Index Terms—Predictive Sampling, Image Sensing, Image Processing, Machine Learning.

I. INTRODUCTION

One of the primary challenges for image sensing and processing comes from the large size of the raw data from image sensors [1]. A high-resolution image sensor produces a large amount of image data, increasing the processing units' computing overhead. The high computing overhead is also linked to the high power consumption of the device. The recent efforts to alleviate this problem can be categorized into hardware methods and software methods. The hardware of image sensing and processing contains pixel circuits, analog-to-digital converters (ADC), and the image signal processor (ISP). Most of the power in an image sensor is consumed by the column-parallel ADCs and digital circuits [2]. Various methods have been proposed to reduce the ADC power such as reducing supply voltage [3], using multi-stage ADCs [4], and applying data compression [5]. However, reducing supply voltage may affect the dynamic range of the sensor. Although Multi-stage ADCs can reduce the power consumption of the image sensor, they still suffer from large data throughput which requires high computing overhead. Data compression needs additional circuits for image compression and reconstruction. On the software side, specialized embedded machine-learning models have been applied to reduce model size for limited hardware resources. For example, [6] proposed a hardware-friendly user-specific machine learning algorithm for edge devices that leverage transfer learning. Standard embedded machine learning solutions have been proposed, such as Tiny Machine Learning (TinyML) [7], [8]. However, these methods do not solve the high computing overhead problem since the image sensor still generates much redundant data for the model.

In this paper, we applied a predictive sampling method for image sensing and processing, which reduces the data throughput and computing overhead for low-power applications. The predictive sampling method was recently proposed for time domain signal sensing and processing systems [9], [10]. It uses the digital value of prior samplings to predict values of the incoming data and only performs digitization when the prediction fails. One advantage of this method is that it can save power consumption by avoiding unnecessary comparison in quantization if the prediction is correct so that the data conversion can be skipped. A more important advantage is that it greatly reduces the output digital data throughput, which can save computing overhead and power consumption of the following digital processing unit. Unlike a more established solution such as compressed sensing [15], dynamic sampling approaches the problem in the pre-processing phase to reduce the data throughput. Compared to other sampling methods such as the adaptive sparse image sampling [16], our sampling method allows for more effective real-time implementation on edge devices.

When applying the predictive sampling method in image sensing and processing, the proposed system scans the entire image and uses the digital value of the neighboring pixels to predict the current pixel value. The scan includes both horizontal row pixel scan and vertical column pixel scan. The goal of these scans is to identify the location and digital values of the pixels that are turning points in each row and column. These pixels are the critical pixels in the image while other pixels may be removed without introducing large distortion of the original image. By doing so the whole image can be compressed to save data throughput. Moreover, the critical pixel locations and values can be directly processed using a low-computing overhead image processing algorithm to save power and memory. While this method can be implemented at the pixel level using a similar method of prediction ADC architecture [2], in this paper, we perform analysis using an existing full digital image to evaluate the performance of the proposed method.

II. SYSTEM DESIGN

A. Predictive Sensing in Image Sensing

The one-dimensional dynamic predictive sampling method can be

Corresponding author: Wei Tang (e-mail: wtang@nmsu.edu).
Associate Editor: Alan Smithee.
Digital Object Identifier 10.1109/LSENS.2017.0000000

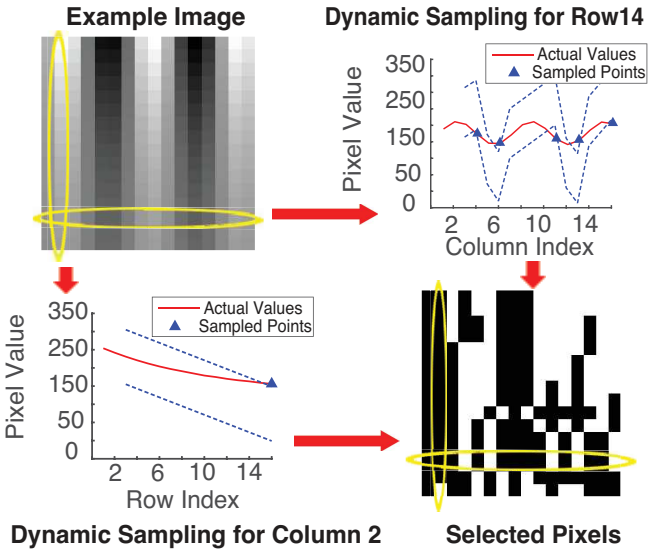

Fig. 1: Predictive sampling method identifies turning points in every row and column. Column 2 and Row 14 are shown as examples. The turning points are selected using the predicted slope of the row or column and two predefined thresholds. The selected pixel is marked in white and the values are recorded. Other pixel values are removed.

implemented in two-dimensional image sensing and processing by considering each row or column as a one-dimensional signal. The overall goal is to locate and record the turning points in each row and column, and then represent each row and column using these points as piece-wise linear wave. If the image is sparse, most of the pixel values are not necessary to be sampled or quantized. only the locations and values of critical pixels need to be recorded. This saves much digital storage space and signal processing overhead, depending on the sparsity of the incoming images. The accuracy of the processing outcome depends on the specific application, which can be adjusted by the error tolerance threshold. Moreover, since the sampling points can represent the information of the image, a low-overhead computing model can use only these sampling points to perform image processing, such as classification.

For example, a grayscale image can be represented by a three-dimensional shape with the X and Y directions referring to the pixel locations and the Z direction for the amplitude value of the pixel. The ranges of the X and Y directions limit the size of the image while the range of the Z direction represents the amplitude. To apply the dynamic predictive sampling method in image sensing, we divide the image into a series of pixel rows and columns. Each row and column is then a vector of pixel amplitude. After applying the dynamic predictive sampling method to each vector, the pixels that represent the amplitude turning points are selected. This can also be done during the sensing phase using prediction-based ADC architecture [2]. The collection of the values of these pixels (X, Y, and Z) is then recorded to represent the image.

Fig. 1 shows an example of the above-mentioned process. The prediction starts at the edge pixels, two consecutive pixels are applied to form a prediction. An error threshold $Delta$ is applied to the process so that a small variation of the amplitude is ignored. If a higher accuracy of the image is required during reconstruction, a smaller $Delta$ value should be applied. For example, a zero $Delta$ keeps all the pixel values. A pixel is marked as a turning point if its value is outside the window formed by the prediction value and the error threshold. In the example image, some pixels are identified as horizontal turning points while some other pixels are identified as vertical turning points. The combination of these two groups is selected as the collection of turning points, which are marked white. The proposed method is different from conventional edge detection since the error threshold is applied to the slope of the prediction instead of the absolute amplitude.

Algorithm 1 Predictive Image Sampling Algorithm

Load image into matrix A
Initialize matrices H and V with zeros and of the same size as A
Set threshold δ to given value
{Process for horizontal pixel selection}
for each row j in image A **do**
  Initialize prediction array P of zeros, length equal to number of columns in A
  Set $P[1]$ and $P[2]$ to the values of the first two pixels in row j
  for each column i from 3 to end in row j **do**
    Predict $P[i] = 2 \times P[i-1] - P[i-2]$
    Define $UTresh = P[i] + \delta$ and $LTresh = P[i] - \delta$
    if $A[j,i] < LTresh$ or $A[j,i] > UTresh$ **then**
      Mark $H[j,i] = 1$
      Update $P[i-1]$ and $P[i]$ to the latest values based on $A[j,i-1]$ and $A[j,i]$
    end if
  end for
end for
{Similar Process for vertical pixel selection}
for each row j in image A **do**
  Initialize prediction array P of zeros, length equal to number of columns in A
  Set $P[1]$ and $P[2]$ to the values of the first two pixels in row j
  for each column i from 3 to end in row j **do**
    Predict $P[i] = 2 \times P[i-1] - P[i-2]$
    Define $UTresh = P[i] + \delta$ and $LTresh = P[i] - \delta$
    if $A[j,i] < LTresh$ or $A[j,i] > UTresh$ **then**
      Mark $V[j,i] = 1$
      Update $P[i-1]$ and $P[i]$ to the latest values based on $A[j,i-1]$ and $A[j,i]$
    end if
  end for
end for
Combine matrices H and V to get matrix T

B. Image Reconstruction

Depending on the application specification and the sparsity of the image, the value of pixels that are not edge or turning points may not be necessary to be stored. While this saves the size of the image file, it is necessary to reconstruct the image using the values of the selected pixels. This can be achieved in two-dimensional (2-D) interpolation, in which the image is reconstructed by neighboring key sampling pixels and forms triangle meshes. The algorithm performs interpolation of scattered data points onto a regular grid and fills the missing data. Depending on the interpolation method, the algorithm creates a series of triangular or sinusoidal meshes that approximate the original image. Fig 2 illustrates the example sampling and reconstruction result of the “Camera Man” image. Fig. 2 (a) to (b) display the key sampling points identified using horizontal and vertical predictive sampling and (c) shows the combined full sample. Fig. 2 (d) shows that with

(a) Horizontal Key Pixels of "Camera Man" (b) Vertical Key Pixels of "Camera Man"

(c) Combined Key Pixels of "Camera Man" (d) Reconstruction Image of "Camera Man"

Fig. 2: Predictive Sampling of the example image "Camera Man" and reconstructed image with the same error threshold.

the collection of the key sampling points, 2-D reconstruction provides a good reconstruction quality.

The error threshold $Delta$ determines the trade-off between the data throughput and the quality of reconstructed image. Figure 3 compares the selected pixels (top row), reconstructed image (middle row), and difference between the original image and the reconstructed image (bottom row) of four $Delta$ values. From the left to the right of Fig. 3, the $Delta$ values are “128", “64", “32", and “16", respectively. The original image is a gray-scale “Camera Man" image with a 8-bit resolution. The value of the pixels are between 0 and 255. As shown in Fig. 3, a higher $Delta$ value results in a lower quality of reconstructed image. The quality of the reconstructed images are quantitatively evaluated using structural similarity index measure (SSIM) values. SSIM is an image quality computation method and metric-based, which assesses reconstruction quality based on the luminance, contrast, and structure of image objects [14], making it much more akin to the human perception and evaluation of image quality. A higher SSIM indicates more similarity between the original image and the reconstructed image. From the left to the right of Fig. 3, the SSIM values are 0.7754, 0.8624, 0.9236, and 0.9560, respectively. The acceptable SSIM of images depends on specific applications.

III. IMAGE PROCESSING EVALUATION RESULTS

A. Datasets and Machine Learning Models

As predictive sampling produces a less data-intensive copy of images containing only the essential pixels within the original, it is possible to perform image processing only using these samples to save computing overhead. To validate this idea, we trained multiple image classification convolutional neural network (CNN) models using datasets of sampled “dog” and “cat” images, a typical image classification problem for model performance assessment. In our testing, We trained three models of varying complexity levels on three different “cat vs dog” datasets, in total nine model for comparison. The datasets selected for the training was a modified version of the Kaggle “Dogs vs. Cats” dataset [11]. The original dataset contains 25,000 images (12,500 “dog" and 12,500 “cat" images). We selected a subset of images from this dataset with 452 ‘dog" and 452 “cat" images. The selected images are pre-processed to grayscale images

(a) $Delta$ = 128, SSIM = 0.7754 (b) $Delta$ = 64, SSIM = 0.8624 (c) $Delta$ = 32, SSIM = 0.9236 (d) $Delta$ = 16, SSIM = 0.9560

Fig. 3: Performance of the proposed method with different error threshold $Delta$: (Top) Selected Pixels (white) using Predictive Sampling; (Middle) Reconstructed image; (Bottom) SSIM map of the reconstructions in contrast with the original image.

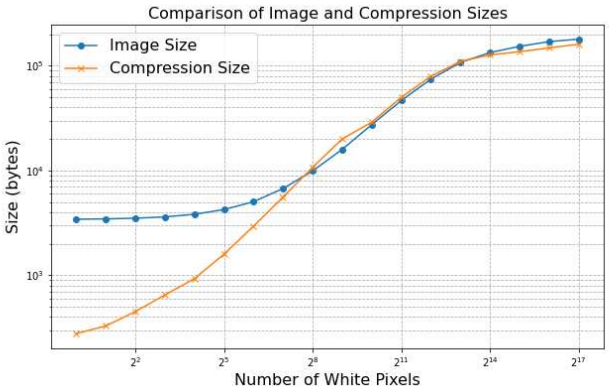

Fig. 4: File size of black and white images with given numbers of selected pixels and their RLC compressed Matlab file equivalent.

with 8-bit resolution for pixel amplitude (0-255). These images are then processed using predictive sampling with a $Delta$ of 126. Therefore, we obtained three datasets as colored images, grayscale images, and predictive sampling images. The files sizes of these datasets are 31.4 MB, 20.8 MB, and 8.51 MB, respectively.

Three sequential CNN models were trained using each of the datasets; Small, Medium, and Large CNNs. All of the models were trained for 20 epochs. The architectural frameworks of these models are 1) *Small and Medium CNNs*: The small and Medium CNNs have similar structures. The only difference between them is the number of kernels in the convolution layers. The small CNNs have half of the number of kernels per convolution layer as the Medium CNNs. They each consist of one input layer, three convolutional layers, a pooling layer, a flattening layer, and a final dense layer with a single neuron for binary classification and ‘Sigmoid’ activation. The models are then compiled with the ‘Adam’ optimizer. 2) *Large CNN*: The structure of the large CNN models includes a larger number of kernels and no zero padding in the convolution layers, as well as two dense layers prior to the final binary classification. It consists of one input layer, four convolutional layers, a pooling layer, two flattening layer, and a final dense layer with a single neuron for binary classification and ‘Sigmoid’ activation. The model is then compiled with the ‘Adam’ optimizer.

B. Classfication Results

The performances of the models are evaluated using the validation datasets. During the training process, the evaluation metrics for the validations are $loss$ and $accuracy$. After the training process, the evaluation metrics are $precision$, $recall$, and $accuracy$. These results are shown in Table 1. The models trained on the sample images are comparable to the gray-scale and color image models in terms of accuracy and inference. In terms of speed, the models trained on the sample images generated using the predictive sampling method had a better inference rate than those of gray-scale and color image models. In terms of accuracy, the predictive sampling models were comparable in the case of the large model and better in the gray-scale and color image models. The main advantage of the predictive sampling method can be seen in the file size of the dataset for training. Compared to other implementations of embedded machine learning for image processing, the proposed predictive sampling method and its associated models offer a good mix of accuracy and inference speed, especially in the small sample model. We compared our results to two other implementations of embedded machine learning as shown in Table 2. While having a slower performance in speed, the predictive sampling models have a higher accuracy. The predictive sampling method has advantages when the input image is sparse. For example, for a 511 by 511 black and white image, the proposed method can reduce the model size only when the number of the selected pixels is under 2^8, which is about 0.1%, as shown in Figure 4.

Table 1: Comparison of the dataset and model size and respective accuracy and inference time.

	Small CNN (21.690 MB)		Medium CNN (43.474 MB)		Large CNN (174.089 MB)	
	.	Inf.	Acc.	Inf.	Acc.	Inf.
Color Image (31.4 MB)	96.9%	35.21 (ms)	98.4%	37.5 (ms)	57.8%	49.38 (ms)
Grayscale Image (20.8 MB)	98.4%	34.77 (ms)	96.9%	38.67 (ms)	60.9%	48.62 (ms)
Predictive Sampled Image (8.51 MB)	96.9%	34.52 (ms)	98.4%	37.23 (ms)	64%	48.21 (ms)

Table 2: Performance Comparison to Similar Solutions in terms of accuracy and inference time.

	Accuracy	Inf. Time
SVM Classifier [12]	92.47%	0.28(ms)
EyeLearn (with SVM) [13]	81.44%	0.16B (FLOPs)
EyeLearn (with 2-Layer MLP) [13]	81.5%	0.16B (FLOPs)
This work: Small CNN (Predictive Sampling)	96.9%	34.52(ms)
This work: Medium CNN (Predictive Sampling)	98.4%	37.23(ms)

IV. CONCLUSION

In this work, we presented a predictive sampling method for selecting key pixels in image sensing and demonstrated the feasibility of using the key pixels for image classification. The proposed method reduces the image size while allowing for effective image reconstruction using only 27.1% of the input data size of the original image. This reduction in image size allows for the reduction in the file sizes of image datasets and the faster training of image classification, which would benefit computer vision implementation of edge devices without cloud connectivity. We used our sample models alongside their color and gray-scale equivalents to train image classification CNNs of different complexity levels to test the validity of the proposed methods. Our results showed that despite the far smaller dataset file size, the sample models had comparable accuracy ratings to that of the color and gray-scale models.

ACKNOWLEDGMENT

This project was partially supported by United States National Science Foundation with grant ECCS-2015573.

REFERENCES

[1] T. Ma, A. Boloor, X. Yang, W. Cao, P. Williams, N. Sun, A. Chakrabarti, and X. Zhang, “Leca: In-sensor learned compressive acquisition for efficient machine vision on the edge,” Proceedings of the 50th Annual International Symposium on Computer Architecture, 2023.

[2] H. Yu, W. Tang, M. Guo, and S. Chen, “A two-step prediction adc architecture for integrated low power image sensors,” IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 64, no. 1, pp. 50–60, 2017.

[3] R. Ozgun, J. Lin, F. Tejada, P. Pouliquen, and A. G. Andreou, “A low-power 8-bit sar adc for a qcif image sensor,” in 2011 IEEE International Symposium of Circuits and Systems (ISCAS), 2011, pp. 841–844.

[4] K. Kitamura, T. Watabe, T. Sawamoto, T. Kosugi, T. Akahori, T. Iida, K. Isobe, T. Watanabe, H. Shimamoto, H. Ohtake, S. Aoyama, S. Kawahito, and N. Egami, “A 33-megapixel 120-frames-per-second 2.5-watt cmos image sensor with column-parallel two-stage cyclic analog-to-digital converters,” IEEE Transactions on Electron Devices, vol. 59, no. 12, pp. 3426–3433, 2012.

[5] N. Katic, M. Hosseini Kamal, M. Kilic, A. Schmid, P. Vandergheynst, and Y. Leblebici, “Column-separated compressive sampling scheme for low power cmos image sensors,” in 2013 IEEE 11th International New Circuits and Systems Conference (NEWCAS), 2013, pp. 1–4.

[6] V. Goyal, R. Das, and V. Bertacco, “Hardware-friendly user-specific machine learning for edge devices,” ACM Trans. Embed. Comput. Syst., vol. 21, no. 5, oct 2022.

[7] H. Liu, Z. Wei, H. Zhang, B. Li, and C. Zhao, “Tiny machine learning (tiny-ml) for efficient channel estimation and signal detection,” IEEE Transactions on Vehicular Technology, vol. 71, no. 6, pp. 6795–6800, 2022.

[8] S. A. R. Zaidi, A. M. Hayajneh, M. Hafeez, and Q. Z. Ahmed, “Unlocking edge intelligence through tiny machine learning (tinyml),” IEEE Access, vol. 10, pp. 100 867–100 877, 2022.

[9] M. Renteria-Pinon, X. Tang, and W. Tang, “Real-time in-sensor slope level-crossing sampling for key sampling points selection for wearable and iot devices,” IEEE Sensors Journal, vol. 23, no. 6, pp. 6233–6242, 2023.

[10] X. Tang, M. Renteria-Pinon, and W. Tang, “Dynamic predictive sampling analog to digital converter for sparse signal sensing,” IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 70, no. 7, pp. 2360–2364, 2023.

[11] W. Cukierski, “Dogs vs. cats,” 2013. [Online]. Available: https://kaggle.com/competitions/dogs-vs-cats

[12] S. Ramadurgam and D. G. Perera, “A systolic array architecture for svm classifier for machine learning on embedded devices,” in 2023 IEEE Fig. 16. File size of 511 by 511 black and white JPG images with given numbers of white pixels and their RLC compressed Matlab file equivalent International Symposium on Circuits and Systems (ISCAS), 2023, pp. 1–5.

[13] M. Shi, A. Lokhande, M. S. Fazli, V. Sharma, Y. Tian, Y. Luo, L. R.Pasquale, T. Elze, M. V. Boland, N. Zebardast, D. S. Friedman, L. Q. Shen, and M. Wang, “Artifact-tolerant clustering-guided contrastive embedding learning for ophthalmic images in glaucoma,” IEEE Journal of Biomedical and Health Informatics, vol. 27, no. 9, pp. 4329–4340, 2023.

[14] Zhou Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," in IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, April 2004, doi: 10.1109/TIP.2003.819861.

[15] M. F. Duarte et al., "Single-pixel imaging via compressive sampling," in IEEE Signal Processing Magazine, vol. 25, no. 2, pp. 83-91, March 2008, doi: 10.1109/MSP.2007.914730.

[16] A. Taimori and F. Marvasti, "Adaptive Sparse Image Sampling and Recovery," in IEEE Transactions on Computational Imaging, vol. 4, no. 3, pp. 311-325, Sept. 2018, doi: 10.1109/TCI.2018.2833625.