Dynamic Hand Gesture Recognition for Wearable Devices with Low Complexity Recurrent Neural Networks

Sungho Shin and Wonyong Sung
Department of Electrical and Computer Engineering
Seoul National University
1, Gwanak-ro, Gwanak-gu, Seoul 08826 Korea
Email: shshin@dsp.snu.ac.kr, wysung@snu.ac.kr

Abstract—Gesture recognition is a very essential technology for many wearable devices. While previous algorithms are mostly based on statistical methods including the hidden Markov model, we develop two dynamic hand gesture recognition techniques using low complexity recurrent neural network (RNN) algorithms. One is based on video signal and employs a combined structure of a convolutional neural network (CNN) and an RNN. The other uses accelerometer data and only requires an RNN. Fixed-point optimization that quantizes most of the weights into two bits is conducted to optimize the amount of memory size for weight storage and reduce the power consumption in hardware and software based implementations.

Keywords—Recurrent Neural Network, Wearable Devices, Quantization, Hand Gesture Recognition, Fixed-point Optimization, Deep Neural Network

I. INTRODUCTION

Recently, many wearable devices have been developed for diverse applications, such as smart watches, Google Glass, and smart bands. Since most wearable devices do not equip keyboards or wide touch screens, it is very necessary to employ speech or gesture recognition technologies. Although speech recognition can be more versatile, the gesture recognition can also be conveniently used for issuing simple commands.

There are several studies and applications that use the hand gesture technology in wearable devices. For example, Google Glass was controlled by hands and feet in [1], smart watches were controlled using flexible force sensors in [2], and SixthSense employed a camera and a projector for interaction with real world [3].

Generally gestures can be classified into static and dynamic ones. Static gestures are usually represented by the hand shapes, while dynamic gestures are described according to hand movements [4]. Gesture recognition can be conducted using signal from a camera or a force sensor. The former needs video processing while the latter analyzes time-varying multi-channel sensor output signal. When a force sensor is used, only the dynamic gesture recognition can be conducted.

Recently, several hand gesture recognition algorithm have been developed [5]–[8]. [5] adopted the correlation between two videos which is called tensor canonical correlation analysis (TCCA). A convolutional neural network (CNN) is used for vision-based static hand gesture recognition for human robot interaction (HRI) [6]. A multimodal gesture detection and recognition is studied using depth video, articulated pose, and audio stream [7]. They applied a CNN, a hidden Markov model (HMM) based speech recognizer, and a bag of word (BoW) to extract the multimodal features. For data fusion and gesture classification, they employed an Elman RNN. In [8], an infrared LED data from Leap Motion Controller was used, where a CNN was applied for feature extraction and an HMM was adopted for time series recognition.

Neural networks are employed to many recognition applications including object detection and speech recognition [9]. However, neural networks demand heavy computation and large memory. For example, a long-short term memory (LSTM) RNN with the unit size of 256 demands a total of approximately 2.1 million (M) weights [10]. For this reason, some of current wearable device applications such as speech recognition operate using servers that employ graphics processing units (GPUs) or multi-core systems consuming quite large power. Thus, it is greatly needed for wearable devices to operate neural network algorithms with only small power.

In this study, we have developed dynamic gesture recognition techniques using fixed-point recurrent neural networks that are suitable for hardware or embedded system based implementations and low-power operation. Two gesture recognition algorithms are implemented; one uses the video signal from a camera and the other utilizes a three-axis accelerometer. Since dynamic gesture recognition needs to analyze the hand-movements, we employ LSTM RNNs. Also, a CNN is attached in front of the RNN for video based hand gesture recognition.

The RNN is optimized to minimize the hardware complexity. In order to minimize the memory size for weight storage, we conduct retrain based fixed-point optimization and successfully reduce most of the word-length into 2 bits [11], [12].

This paper is organized as follows. In Section II the proposed hand gesture recognition models and quantization procedure are given. Experimental results are provided in Section III and concluding remarks follow in Section IV.

II. FIXED-POINT RNN OPTIMIZATION

We employ two different kinds of dynamic hand gesture dataset. One is based on the image sequence [5], and the other is the 3-axis acceleration data [13]. Also, a fixed-point optimization scheme for these algorithms is explained.
### A. Image Sequence Based Dynamic Hand Gesture Recognition

For the image sequence based dynamic hand gesture recognition task, we employed a CNN-LSTM RNN structure. The three layer CNN is similar to the one proposed by [14]. The specific network structure is depicted in Fig. 1 which shows three convolution and pooling layers followed by an RNN layer. The input layer consists of 3072 (=3×32×32) linear units for handling the 32 by 32 input images with RGB channels. The first and second convolution layers have 32 feature maps and the third convolution layer has 64 feature maps. These three layers have the same convolution kernel size, which is 5×5. The three pooling layers employ 2×2 overlapping max pooling. Thus, the CNN demands 79.2 kilo weights. Rectified linear units are adopted as for the activation functions. We employ the RNN layer to analyze gesture’s temporal relation. The LSTM RNN can remember quite long past information in the sequence. As a result, HMM networks are not needed in this recognition model. The output layer consists of 9 softmax units which correspond to 9 target gesture behaviors. The total number of weights for the LSTM RNN is approximately 99,456 kilo weights. Therefore, a total of 0.714 MB (79.2 K and 99.456 K weights for CNN and RNN, respectively) memory space is required for the network model in a 32-bit floating-point format.

### B. Acceleration Data Sequence Based Dynamic Hand Gesture Recognition

The acceleration data sequence based dynamic hand gesture recognition model also employs the LSTM RNN structure. The standard LSTM uses three gates which are called the input gate, forget gate and output gate that can access and modify the memory cells. The activation functions for these three gates are the logistic sigmoid, and the input and output layers of the LSTM employ the hyperbolic tangent activation functions. This algorithm applies the acceleration data directly to the RNN. Therefore, this application is much simpler than the image sequence based dynamic hand gesture recognition. The input layer contains 3 linear units to receive the 3-axis acceleration data. One LSTM hidden layer with size of 128 is used, and the output layer consists of 8 softmax units which correspond to 8 target gesture movements. An LSTM layer with \( N \) units demands a total of \( 4N^2 + 4NM + 7N \) weights where \( M \) is the previous layer size [10]. Therefore, the total number of weights is approximately 69 K, and as a result 276 KB memory space is needed for the network model when a floating-point format is used.

### C. Retrain-Based Weight and Signal Quantization

The quantization effects of signals or weights depend on a signal flow graph, and the influence of quantization can be represented as the sensitivity [15]. The weights and signals in each layer are grouped and each group employs the same quantization step size \( \Delta \). To optimize \( \Delta \), we adopt L2 error minimization criteria as suggested in [11], [16].

Based on the quantization step size \( \Delta \), sensitivity analysis for weights and signals is conducted layerwisely. Fig. 1 shows the weight and signal grouping results. In this figure, ‘In-C1’ is the first weight group between the input layer and the first convolution layer, ‘S3-L1’ is the fourth weight group between the last pooling layer and the LSTM layer. ‘C1’ is the signal group of the first CNN layer, and ‘L1’ is the signal group of the LSTM layer.

Since direct quantization does not show good performance, retraining on the quantization domain is performed. The RNN version of the retraining algorithm is introduced in [12].

In our target networks, we use three different types of activation functions (logistic sigmoid, hyperbolic tangent and rectified linear unit). The output range of the logistic sigmoid function is between 0 and 1, that of the hyperbolic tangent is -1 and 1, and that of the rectified linear unit is theoretically 0 and \( \infty \). Thus, the output signals for the logistic sigmoid and hyperbolic tangent activation functions are quantized with a fixed size of \( \Delta \). However, the output value of the rectified linear unit can be unbounded. Therefore, the quantization step size \( \Delta \) needs to be calculated in a similar method with the weights quantizer. The output signals of the rectified linear units are saved all over the training set to compute the proper quantization step size \( \Delta \) with L2 error minimization.

### III. EXPERIMENTAL RESULTS

The proposed algorithms are evaluated using two datasets obtained from wearable devices. One is the image based hand gesture recognition dataset and the other is the acceleration dataset from a 3-axis accelerometer. Advanced training techniques such as early stopping, adaptive learning rate, and Nesterov momentum are employed [17], [18].

### A. Image Based Dynamic Hand Gesture Recognition

Image based dynamic hand gesture recognition experiments were performed on the Cambridge-Gesture data base [5]. The dataset consists of 900 image sequences of 9 gesture classes in QVGA (320 by 240), which are defined by 3 primitive hand shapes (flat, spread, and V-shape) and 3 primitive motions (left, right, and contract). Therefore, the target task for this data set is to classify different shapes as well as different motions simultaneously. Each class contains 100 image sequences (5 different illuminations \( \times 10 \) arbitrary motions \( \times 2 \) subjects). The dataset was divided into 60% for the training (540 sequences), 20% for the validation (180 sequences), and 20% for the test (180 sequences) randomly. The ratios of the class labels are the same for the three sets.
TABLE I. LAYERWISE SENSITIVITY ANALYSIS RESULTS FOR SIGNAL GROUPS IN THE IMAGE SEQUENCE BASED MODEL. EACH LAYERS IS QUANTIZED IN TWO BITS FOR SENSITIVITY ANALYSIS. THE NUMBERS IN THE TABLE REPRESENT THE MISS CLASSIFICATION RATE (%) OF THE TEST SET. ‘D’ MEANS DIRECT QUANTIZATION RESULTS AND ‘R’ REPRESENTS THE RESULTS AFTER RETRAINING.

|       | In | C1  | S1  | C2  | S2  | C3  | S3  | L1  | All |
|-------|----|-----|-----|-----|-----|-----|-----|-----|-----|
|       | D  | 25.97 | 28.41 | 27.25 | 31.80 | 29.61 | 23.49 | 23.68 | 58.24 |
|       | R  | 24.31 | 20.15 | 24.26 | 23.49 | 22.56 | 23.86 | 22.90 | 24.40 |

The network is trained using Fractal RNN library with training parameters that are 64 forward steps and 64 backward steps with 8 streams [19]. Initial learning rate was $10^{-5}$ and the learning rate is decreased until $10^{-8}$ during the training. Momentum was 0.9 and AdaDelta was adopted for weights updating [20]. We tried to find out the proper input size of the CNN with 64 by 64, 32 by 32, and 16 by 16 images. Their floating point classification error rates were 22.46%, 22.79%, and 73.70% respectively. Therefore, the 32 by 32 image size is selected as the input image dimension. The network demands approximately 178.656 kilo weights. All experiments are repeated five times to consider their noise effects.

Fig. 3 shows the layerwise fixed-point sensitivity analysis for the weight groups. The original miss classification rate was 22.79% for the test set. The results indicate that the most sensitive weight groups are ‘In-C1’ and ‘L1-Out’. However, the final fixed-point network employs only 2 bits for all the weight groups because their sensitivity difference is small. TABLE I depicts the results of the signal group sensitivity analysis. All the signal groups also employ only two bits after retraining. Note that some layers even show better performance when compared to the floating point results. We obtained a fully quantized network with these two sensitivity results. All the weight and signal groups were quantized by using only two bits, and the miss rate was 25.04%. With this quantization, the memory space saved is 93.75% (=30/32) when compared to a floating-point implementation. The total number of multiplications for each layer is 56,448 M (C1), 76.8 M (C2), 1.536 M (C3), 1.966 M (L1) and 34.56 K (Out), respectively, for real-time operation (30 Hz). The memory space needed is 44.625 KB. Therefore, our model can efficiently be implemented in embedded systems such as Cortex-A9 (128 KB - 8 MB L2 cache), since the whole weights memory can be stored in the on-chip L2 cache.

B. Accelerometer Based Dynamic Hand Gesture Recognition

An accelerometer based dynamic hand gesture recognition model was trained using the SmartWatch Gestures Dataset [13]. The data set has been collected to evaluate several gesture recognition algorithms for interacting with mobile applications using arm gestures. Eight different users performed twenty repetition of twenty different gestures for a total of 3200 sequences. Each sequence contains acceleration data from the 3-axis accelerometer of a first generation Sony SmartWatch. Original dataset contains 20 motions, but eight motions which are depicted in Fig. 2 (b) are enough as a wearable device controller. The training, validation, and test sets are divided randomly into 50%, 20%, and 30% respectively.

The RNN training method is the same with [19]. Initial learning rate was $10^{-5}$ and it is decreased until $10^{-7}$ during the training procedure. Momentum was 0.9 and AdaDelta was employed for weight updating. We tried to find out the proper network size of the LSTM layer. The network sizes of 32, 64, 128 and 256 were considered. Since the entire number of the dataset is too small to training the LSTM RNN, the test set noise was very large. Therefore we conduct the experiments 10 times for each network size. The floating point training results were 36.02±24.40% (mean±standard deviation), 22.26±4.00%, 18.29±4.86%, and 16.68±5.72%.
for the 32, 64, 128 and 256 network size, respectively. We chose the LSTM layer with the size of 128, and the floating-point weights that show best error rate on the test set, which was 11.43%.

TABLE II shows the layerwise sensitivity analysis results for weights and signals. The floating point classification error rate was 11.43%. Since the training set is too small for training the LSTM RNN, overfitting was observed when retraining the network in the quantization domain. The signal sensitivity analysis result shows that the ‘L1’ layer group does not yield good performance with only two bits quantization levels. It needs three or more bits to obtain acceptable results.

We next try fixed-point optimization of all signals and weights using the sensitivity analysis results. We applied two bits to all weight and signal groups and its miss classification rate was 32.77%. Therefore, we assigned more bits to the ‘L1’ layer signal group and obtained 28.69% in three bits and 11.43% in four bits. The whole memory space can be reduced by 93.75% when compared to a floating point implementation, since the memory space for weights is only 17.25 KB. The total number of multiplications was 690 K for real time operation (10 Hz). Our model can be efficiently implemented in embedded systems, since the whole weights can be stored in the cache memory.

IV. CONCLUDING REMARKS

This work investigates the fixed-point implementation of LSTM RNNs for dynamic hand gesture recognition using two different datasets. The RNN based implementations show good results and can be improved further by using more training data. The retrain-based fixed-point optimization greatly reduces the word length of the weights and signals. By this optimization, the required memory space for weights can be reduced to only 6.25% compared to floating-point implementations. The optimized fixed-point network can show much better energy performance in embedded implementations because the reduced memory size enables purely on-chip memory based operations.

ACKNOWLEDGMENT

This work was supported in part by the Brain Korea 21 Plus Project and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2015R1A2A1A10056051).

REFERENCES

[1] Z. Lv, L. Feng, H. Li, and S. Feng, “Hand-free motion interaction on google glass,” in SIGGRAPH Asia 2014 Mobile Graphics and Interactive Applications. ACM, 2014, p. 21.

[2] E. Morongi, L. Angelini, A. Adami, D. Lalanne, L. Lorenzelli, and E. Mugellini, “A smart watch with embedded sensors to recognize objects, grasps and forearm gestures,” Procedia Engineering, vol. 41, pp. 1169–1175, 2012.

[3] P. Mistry and P. Maes, “Sixthsense: a wearable gestural interface,” in ACM SIGGRAPH ASIA 2009 Sketches. ACM, 2009, p. 11.

[4] H. Hasan and S. Abdul-Kareem, “Static hand gesture recognition using neural networks,” Artificial Intelligence Review, vol. 41, no. 2, pp. 147–181, 2014.

[5] T.-K. Kim and R. Cipolla, “Canonical correlation analysis of video volume tensors for action categorization and detection,” Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 31, no. 8, pp. 1415–1428, 2009.

[6] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Cireşan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L. M. Gambardella, “Max-pooling convolutional neural networks for vision-based hand gesture recognition,” in Signal and Image Processing Applications (ICSIPA), 2011 IEEE International Conference on. IEEE, 2011, pp. 342–347.

[7] N. Neverova, C. Wolf, G. Paci, G. Sommavilla, G. Taylor, and F. Nebout, “A multi-scale approach to gesture detection and recognition,” in Proceedings of the IEEE International Conference on Computer Vision Workshops, 2013, pp. 484–491.

[8] R. McCartney, J. Yuan, and H. Bischof, “Gesture recognition with the leap motion controller,” in Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing, 2015, p. 3.

[9] K. Hwang and W. Sung, “Character-level incremental speech recognition with recurrent neural networks,” in Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on. IEEE, 2016.

[10] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, “Learning precise timing with LSTM recurrent networks,” The Journal of Machine Learning Research, vol. 3, pp. 115–143, 2003.

[11] K. Hwang and W. Sung, “Fixed-point feedforward deep neural network design using weights +1, 0, and -1,” in Signal Processing Systems (SiPS), 2014 IEEE Workshop on. IEEE, 2014, pp. 1–6.

[12] S. Shin, K. Hwang, and W. Sung, “Fixed-point performance analysis of recurrent neural networks,” in Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on. IEEE, 2016.

[13] G. Costante, L. Porzi, O. Lanz, P. Valigi, and E. Ricci, “Personalizing a smartwatch-based gesture interface with transfer learning,” in Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European. IEEE, 2014, pp. 2530–2534.

[14] A. Krizhevsky, (2012) cuda-convnet. [Online]. Available: http://www.psfk.com/report/future-of-wearable-tech

[15] W. Sung and K.-I. Kum, “Simulation-based word-length optimization method for fixed-point digital signal processing systems,” Signal Processing, IEEE Transactions on, vol. 43, no. 12, pp. 3087–3090, 1995.

[16] S. Anwar, K. Hwang, and W. Sung, “Fixed point optimization of deep convolutional neural networks for object recognition,” in Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on. IEEE, 2015, pp. 1131–1135.

[17] R. A. Jacobs, “Increased rates of convergence through learning rate adaptation,” Neural networks, vol. 1, no. 4, pp. 295–307, 1988.

[18] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” in Proceedings of the 30th international conference on machine learning (ICML-13), 2013, pp. 1139–1147.

[19] K. Hwang and W. Sung, “Single stream parallelization of generalized LSTM-like RNNs on a GPU,” in Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on. IEEE, 2015, pp. 1047–1051.

[20] M. D. Zeiler, “Adadelta: An adaptive learning rate method,” arXiv preprint arXiv:1212.5701, 2012.