The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Quantized Memory-Augmented Neural Networks

Seongsik Park,[1] Seijoon Kim,[1] Seil Lee,[1] Ho Bae,[2] Sungroh Yoon[1,2]
[1]Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Korea
[2]Interdisciplinary Program in Bioinformatics, Seoul National University, Seoul 08826, Korea
sryoon@snu.ac.kr

Abstract

Memory-augmented neural networks (MANNs) refer to a class of neural network models equipped with external memory (such as neural Turing machines and memory networks). These neural networks outperform conventional recurrent neural networks (RNNs) in terms of learning long-term dependency, allowing them to solve intriguing AI tasks that would otherwise be hard to address. This paper concerns the problem of quantizing MANNs. Quantization is known to be effective when we deploy deep models on embedded systems with limited resources. Furthermore, quantization can substantially reduce the energy consumption of the inference procedure. These benefits justify recent developments of quantized multilayer perceptrons, convolutional networks, and RNNs. However, no prior work has reported the successful quantization of MANNs. The in-depth analysis presented here reveals various challenges that do not appear in the quantization of the other networks. Without addressing them properly, quantized MANNs would normally suffer from excessive quantization error which leads to degraded performance. In this paper, we identify memory addressing (specifically, content-based addressing) as the main reason for the performance degradation and propose a robust quantization method for MANNs to address the challenge. In our experiments, we achieved a computation-energy gain of 22× with 8-bit fixed-point and binary quantization compared to the floating-point implementation. Measured on the bAbI dataset, the resulting model, named the quantized MANN (Q-MANN), improved the error rate by 46% and 30% with 8-bit fixed-point and binary quantization, respectively, compared to the MANN quantized using conventional techniques.

Introduction

In recent years, Memory-Augmented Neural Networks (MANNs), which couple the external memory and the neural network, have been proposed for improving the learning capability of the neural network. MANNs, including neural Turing machines (Graves, Wayne, and Danihelka 2014), memory networks (Weston, Chopra, and Bordes 2014), and differentiable neural computers (Graves et al. 2016), can handle complex problems such as long-sequence tasks and Question and Answer (QnA).

Table 1: Computation-energy consumption from (Horowitz 2014)

Type	Arithmetic operation	Bit	Energy (pJ)	Gain[a]
Fixed point	add	8	0.03	123.3×
	mult	8	0.2	18.5×
Floating point	add	32	0.9	4.1×
	mult	32	3.7	1.0×

[a] compared with 32-bit floating-point mult

Although various types of neural networks have shown promising results in many applications, they demand tremendous amounts of computational energy (Silver et al. 2016). This has been the biggest obstacle for employing deep learning on embedded systems with limited hardware resources.

Previous work overcame this challenge by introducing the concept of quantization (e.g., fixed-point and binary) to deep learning (Courbariaux, Bengio, and David 2014; Lin, Talathi, and Annapureddy 2015; Gupta et al. 2015; Hubara et al. 2016; Courbariaux, Bengio, and David 2015; Courbariaux et al. 2016; Rastegari et al. 2016; Zhou et al. 2016; Tang, Hua, and Wang 2017).

Quantization can avoid floating-point operations which consume considerable computing energy (Horowitz 2014). As shown in Table 1, an 8-bit fixed-point addition can obtain a computation-energy gain of 123.3× over a 32-bit floating-point multiplication (Supplementary[1] Table 3).

Typically, low-power and real-time processing are recommended features in a limited-resource environment. NVIDIA has recently introduced TensorRT (Allison et al. 2017), a deep learning inference optimizer for low-power and real-time processing with 8-bit or 16-bit representation. For the reasons mentioned above, quantization has become a critical process to employ deep learning in a limited-resource environment.

In this paper, we applied fixed-point and binary quantization to a conventional MANN to enable it to be employed in a limited-resource environment. We evaluated its feasibility by training and inference with fixed-point quantized param-

[1]Supplementary is available at http://data.snu.ac.kr/pub/Q-MANN/AAAI2017.suppl.pdf

eters and activations, and then further extended this to binary representation. According to our experimental results, the application of 8-bit fixed-point quantization increased the test error rate by more than 160%. This was a significant amount of degradation compared to that of the initial study of fixed-point quantization on CNN (Lin and Talathi 2016).

This prompted us to overcome the limitations of training and inference with quantized parameters and activations in a MANN. We started by theoretically analyzing the effect of the quantization error on the MANN using content-based addressing with cosine similarity. We verified our analysis and revealed that a conventional MANN was susceptible to quantization error through various experiments.

Based on our detailed analysis and experiments, we proposed quantized MANN (Q-MANN) that employs content-based addressing with a Hamming similarity and several techniques to enhance its performance. Our result showed that Q-MANN had a lower test error rate than conventional MANN when both 8-bit fixed-point and binary quantization were applied. The contributions of this paper can be summarized as follows:

- We first attempted to train a conventional MANN with fixed-point and binary quantized parameters and activations, which could be the basis for subsequent research.
- We theoretically analyzed the reasons for the poor training result when applying quantization to the MANN and verified our analysis through various experiments.
- We proposed Q-MANN that could improve both the robustness for the quantization error and the learning performance when a small bit width of quantization was applied.

Related Work

Memory-Augmented Neural Networks

A MANN consists of two main components: a memory controller and external memory. It can learn how to read from and write to the memory through data. Thus, memory addressing plays a key role in training and inference.

There could be several types of addressing method depending on the interpretation of MANN. However, in this paper, we referred to MANN as neural networks represented by neural Turing machines (Graves, Wayne, and Danihelka 2014), differentiable neural computers (Graves et al. 2016) and memory networks (Weston, Chopra, and Bordes 2014). Thus, we focused on content-based addressing which is general addressing method of those neural networks.

Content-based addressing C is defined as

$$C(M,k)[i] = \frac{\exp\{S(M_i,k)\}}{\sum_{j=1}^{L}\exp\{S(M_j,k)\}}, \tag{1}$$

where S is the similarity between the memory element M_i and the key vector k, and L is the number of memory elements. In a conventional MANN, a cosine similarity is used as a similarity measure S between the memory element M_i and the key vector k, and a softmax function is used as a normalization function. Since cosine similarity and softmax are differentiable, MANN is end-to-end trainable using SGD and gradient back-propagation.

However, this approach is not suitable in a limited-resource environment because the cosine similarity that features multiplication, division, and square root computation require tremendous amounts of hardware resources and computational energy. In this regard, using a dot product instead of cosine similarity as in (Sukhbaatar et al. 2015) is much more efficient in a limited-resource environment.

Fixed-point and binary quantization

Fixed-point quantization is a popular choice because it hardly has any quantization overhead and is seamlessly compatible with conventional hardware. Given a floating point u, the fixed point $\hat{u}$ can be expressed as

$$\hat{u} = S_{\hat{u}} 2^{-FRAC} \sum_{k=0}^{n-2} 2^k \hat{u}_k, \tag{2}$$

where n is the bit width, $S_{\hat{u}}$ is the sign, and $FRAC$ is the fraction bit of fixed point $\hat{u}$. The fixed point is composed of a sign, integer, and fraction and each bit width can be denoted as 1, IWL, and $FRAC$. Q-format (Q$_{IWL.FRAC}$) was used to represent a fixed point in this paper.

The quantization error $\epsilon_{\hat{u}}$ is defined as depending on the occurrence of fixed-point overflow,

$$|\epsilon_{\hat{u}}| < \begin{cases} 2^{-FRAC} & \text{if } |u| < 2^{IWL} \\ |2^{IWL} - |u|| & \text{if } |u| \geq 2^{IWL} \end{cases}. \tag{3}$$

More details of the fixed point are provided in Supplementary. As shown in Equation 3, when a fixed-point overflow occurs, the quantization error becomes much larger and this can be a major cause of the degradation of the learning performance of a MANN.

As shown in Table 1, an energy gain of 18.5× could be accomplished through fixed-point quantization by converting 32-bit floating-point arithmetic operations to 8-bit fixed-point. Moreover, we could achieve a 123.3× energy gain by replacing 32-bit floating-point multiplication with 8-bit fixed-point addition through fixed-point and binary quantization. These two approaches allowed us to deploy a MANN in a limited-resource environment.

Fixed-point and binary quantization on deep learning

To date, most of the research on fixed-point and binary quantization of deep learning has focused on MLP, CNN, and RNN (Hubara et al. 2016). Especially, quantization on CNN has been extended beyond fixed-point quantization to binary quantization (Courbariaux, Bengio, and David 2015; Courbariaux et al. 2016; Rastegari et al. 2016; Tang, Hua, and Wang 2017).

Initial studies on fixed-point quantization of CNN include (Gupta et al. 2015) and (Lin and Talathi 2016). By applying 16-bit fixed-point quantization and using stochastic rounding, (Gupta et al. 2015) could obtain a similar result to that of a floating-point on the CIFAR-10 dataset. (Lin and Talathi

2016) confirmed the feasibility of fixed-point quantization on CNN using fewer bits by applying 4-/8-bit fixed-point quantization to CNN.

Based on the initial research, binary quantization on CNN has been studied beyond fixed-point quantization. In BinaryConnect (Courbariaux, Bengio, and David 2015), binary quantization was applied to the parameters in CNN. It showed a comparable result to floating point on the MNIST, CIFAR-10, and SVHN datasets. Through binary quantization, they were able to reduce the computation-energy consumption by replacing multiplication with addition.

BinaryNet (Courbariaux et al. 2016) and XNOR-Net (Rastegari et al. 2016) applied binary quantization of parameters and activations to CNN. The binarization method was the same as that of BinaryConnect, and the estimated gradient was used for gradient back-propagation. Multiplication was replaced with XNOR and addition, which allowed training and inference with less computation-energy consumption than BinaryConnect.

BinaryNet achieved similar results to BinaryConnect on the MNIST and CIFAR-10 datasets. XNOR-Net used a scale factor and adjusted the position of the layers to compensate for the loss of information caused by binarization. As a result, binary quantization could be applied to CNN on the ImageNet dataset with an increase in the error rate of approximately 50% compared to the use of floating point. DoReFa-Net (Zhou et al. 2016) with a quantized gradient showed a lower error rate than XNOR-Net on the ImageNet dataset when the parameters, activations, and gradient were 1-, 2-, and 4-bit, respectively.

The most recently published research (Tang, Hua, and Wang 2017) could improve the learning performance of binarized CNN by using a low learning rate, PReLU instead of a scale factor, and newly proposed regularizer for binarization. They obtained an increase in the error rate of about 20% compared to the using floating point on the ImageNet dataset, which is the lowest error rate thus far.

(Ott et al. 2016) mainly focused on binary and ternary quantization on RNN. They applied quantization to parameters of vanilla RNN, LSTM, and GRU. In their experiments, they obtained a comparable result to those using floating point, but only when the ternarization was applied.

In-Depth Analysis of the Quantization Problem on a MANN

Fixed-point and binary quantization on a MANN

To the best of our knowledge, this work is the first attempt to apply fixed-point and binary quantization to a MANN. Our method involved applying the approaches that are conventionally used to achieve fixed-point and binary quantization in CNNs to a MANN. Our approach was similar to that in (Gupta et al. 2015; Courbariaux, Bengio, and David 2015) and we analyzed the feasibility of applying fixed-point and binary quantization to a MANN. As the early stage research of quantization on CNN, we excluded the last output layer and gradient from quantization and focused on the activations and parameters instead.

We followed the conventional approach to fixed-point quantization by quantizing the parameters, activations, and memory elements to 8-bit fixed point. For binary quantization, we set the activations as binary, and the parameters and memory elements as 8-bit fixed point. We binarized the activations instead of the parameters since the input format (e.g., Bag-of-Words) of MANN can be easily binarized and the parameter can directly affect the memory element significantly. In those respects, our binarization method is different from (Courbariaux, Bengio, and David 2015), but it is similar in that it can reduce computation-energy consumption by replacing multiplication with addition.

Our experimental results indicated that a conventional MANN increased the error rate by over 160% when 8-bit fixed-point quantization was applied. Compared with the result obtained during the initial stage research of quantization on CNN (Lin and Talathi 2016), which had an increased error rate of 40%, the learning performance achieved by applying quantization to the conventional MANN was significantly degraded. In this section, we thoroughly investigate why the quantization affected serious performance degradation in the conventional MANN. Based on the analysis, our proposed approach to solving the stated problem is presented in the later section.

Analysis of the effect of the quantization error on content-based addressing

One of the major causes of performance degradation is content-based addressing which is an external memory addressing mechanism. Content-based addressing typically consists of a cosine similarity (i.e., dot product) and a softmax function (Equation 1).

The distribution of vector similarity used in content-based addressing of a conventional MANN is depicted in Figure 1. As the training progressed, the similarity of the related vectors became larger and that of the less relevant vectors became smaller. Thus, the more training progressed, the wider the distribution of similarity became. This training tendency of the vector similarity was a major obstacle to applying fixed-point quantization to the conventional MANN.

We analyzed the effect of the quantization error on the distribution of the similarity by calculating the similarity $\hat{Z}$,

$$\hat{Z} \approx Z + \sum (u_i \epsilon_{\hat{v}_i} + v_i \epsilon_{\hat{u}_i}), \tag{4}$$

where Z is the similarity of floating-point vector, $\hat{u}_i$ and $\hat{v}_i$ are elements of the quantized input vectors (Supplementary Equation 10). As shown in Equation 4, the influence of the quantization error of the input vectors $\epsilon_{\hat{u}_i}$ and $\epsilon_{\hat{v}_i}$ on the error of similarity measure $\epsilon_{\hat{Z}}$ was proportional to the sum of the products of the input vectors and the quantization error $\sum (u_i \epsilon_{\hat{v}_i} + v_i \epsilon_{\hat{u}_i})$. As mentioned earlier, since the distribution of similarity became wider as the training progressed, the quantization error of the input vectors $\epsilon_{\hat{u}_i}$ and $\epsilon_{\hat{v}_i}$ caused the distribution of similarity to become much wider.

In order to examine the effect of the fixed-point quantization error on the distribution of similarity, we set IWL to 5 and measured the distribution of similarity while decreasing $FRAC$ to 7, 4, and 1, respectively (Figures 1A,

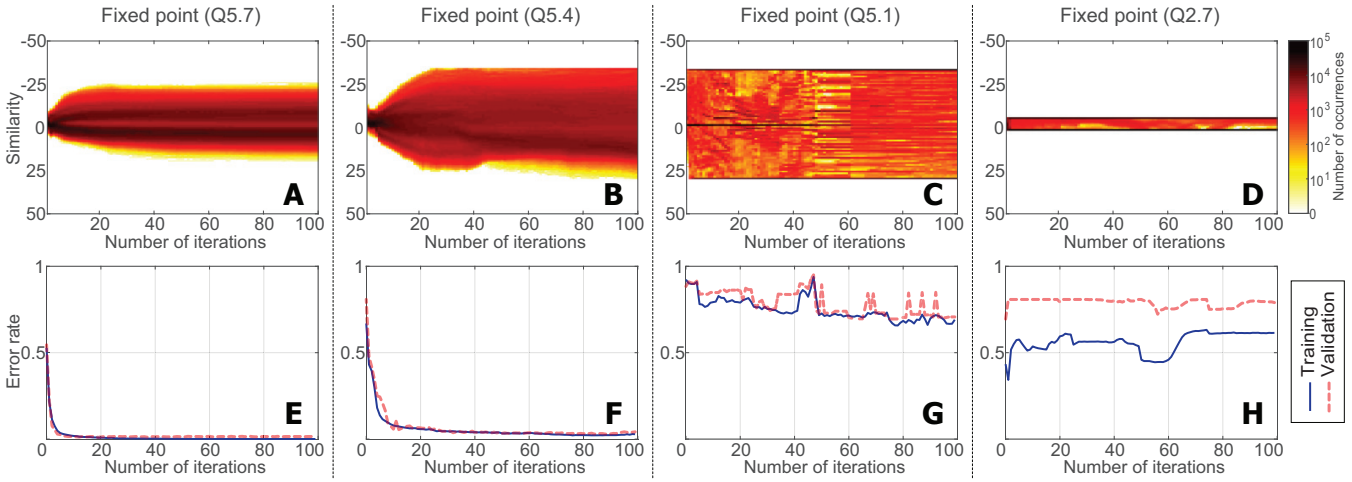

Figure 1: The results for a conventional MANN on the bAbI dataset (task8), (A)-(D): the distribution of the similarity measure, (E)-(H): the error rate of training and validation. The results of measuring the distribution of the similarity by setting IWL to 5 and reducing $FRAC$ to 7, 4, and 1 (A, B, and C, respectively) showed an increase in the width of the distribution as $FRAC$ decreased. In addition, the width of the distribution became wider as the training progressed. The increase of the width caused the fixed-point overflow because a fixed-point representation could indicate a value in a limited range ($< |2^{IWL}|$). If the overflow occurs frequently as (C) and (D), the error rate of training and validation increased considerably as (G) and (H). Even if the same bit width was used for the fixed-point quantization as (B) and (D), the learning performance varied greatly, as (F) and (H), according to the width of IWL and $FRAC$.

1B, and 1C). Since the fixed-point quantization error is inversely proportional to the $FRAC$ (Equation 3), the distribution became wider as the $FRAC$ decreased. The wide distribution of similarity due to the quantization error incurred fixed-point overflow (Figure 1C), which made conventional MANN vulnerable to a quantization error (Figure 1G).

To investigate the effect of the error of similarity measure on the learning performance of a conventional MANN, we analyzed the effect of the input quantization error $\epsilon_{\hat{z}_i}$ on the output of a softmax $\hat{y}_i$ which is a typical normalization function in content-based addressing:

$$\hat{y}_i \leq \exp(\epsilon_{max}) y_i, \tag{5}$$

where y_i is the floating-point output of softmax and ϵ_{max} is the maximum of the input quantization error (Supplementary Equation 11). Assume that index i is referred to as the index of ϵ_{max} among the quantization error $\epsilon_{\hat{z}_i}$ of quantized input $\hat{z}_i$ in softmax. Then, the output of softmax $\hat{y}_i$ would be proportional to $\exp(\epsilon_{max})$. Therefore, the output error $\epsilon_{\hat{y}_i}$ would also be proportional to $\exp(\epsilon_{max})$.

This finding indicated that when softmax is used as a normalization function it is exponentially influenced by the input quantization error. In content-based addressing, the vector similarity Z was used as input of the softmax normalization function. Thus, the error of vector similarity $\epsilon_{\hat{Z}}$ exponentially affected the output error of softmax $\epsilon_{\hat{y}_i}$. Consequently, the error of vector similarity $\epsilon_{\hat{Z}}$ significantly degraded the learning performance of conventional MANN.

Depending on whether fixed-point overflow occurs or not, the fixed-point quantization error varies greatly according to Equation 3. If there were endurable overflows (Figure 1B), the learning tendency was not significantly different (Figures 1A and 3B), although the error rate of applying a fixed-point quantization to a conventional MANN was higher than that of the floating-point representation (Figures 1E and 3F). However, the occurrence of many overflows and an increase in the fixed-point error of the similarity measure (Figure 1C), caused the learning performance to decrease drastically (Figure 1G).

A comparison of our results (Figures 1F and 1H) with fixed-point quantization using the same bit width but different configurations (Q5.4 and Q2.7), enabled us to observe the degradation of learning performance caused by the fixed-point overflow. Although the same bit width was used, there were many more overflows in the case of Q2.7 (Figure 1D), which degraded the learning performance significantly (Figure 1H).

Hence, the fixed-point overflow had a great influence on the learning performance of MANN. The overflow of vector similarity in fixed-point quantization where limited bit width is available can be prevented by increasing IWL but this leads to the use of a smaller $FRAC$. As mentioned earlier, the increased quantization error due to the small $FRAC$ caused fixed-point overflow of the similarity measure as training progressed (Figure 1C). Thus, cosine similarity (i.e., dot product) is not suitable for fixed-point quantization, which uses a small number of bits with a limited numerical representation range.

Quantized MANN (Q-MANN)

Architecture

In this paper, we proposed Q-MANN to overcome the aforementioned disadvantage of conventional MANN, which is

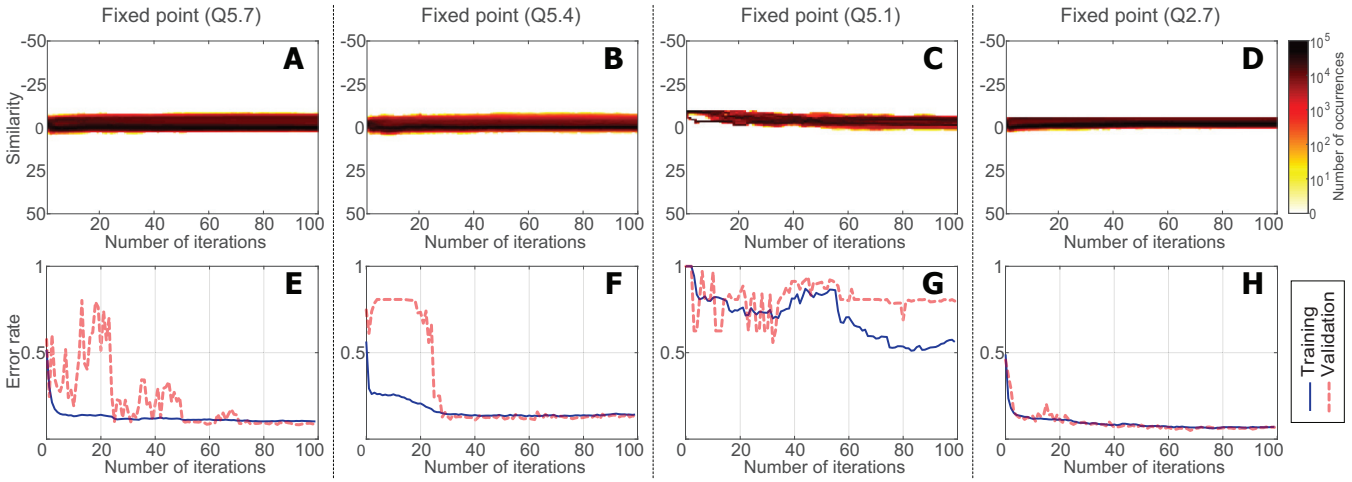

Figure 2: The results for Q-MANN on the bAbI dataset (task8), (A)-(D): the distribution of the similarity measure, (E)-(H): the error rate of training and validation. The results of measuring the distribution of the similarity by setting IWL to 5 and reducing $FRAC$ to 7, 4, and 1 (A, B, and C, respectively), indicated that there was a little difference among the width of the distributions. Even though there was no significant difference in the distribution of the width, the similarity measure became more accurate, and the learning performance was improved as $FRAC$ increases with the same IWL (E, F, and G). In the experiments having the same $FRAC$, the learning performance was improved when we used the distribution-optimized IWL as (H) ($IWL = 2$), even though it was a smaller IWL than that of the other (E) ($IWL = 5$). In the case of using the same bit width (F and H), the results showed a lower error rate of training and validation with the distribution-optimized IWL and $FRAC$ (Q2.7) as (H) compared to (F).

vulnerable to quantization error. As we have seen, the fixed-point overflow in content-based addressing with cosine similarity significantly degraded the learning performance of conventional MANN. Our solution to this problem was using bounded similarity (e.g., Hamming similarity) instead of unbounded similarity (e.g., cosine similarity) in content-based addressing. Bounded similarity could prevent the fixed-point overflow and improve the learning performance when applying a fixed-point quantization. The proposed similarity measure S_H is defined as

$$S_H(\hat{U}, \hat{V}) = \sum_i S_{\hat{u}_i} S_{\hat{v}_i} \sum_{k=0}^{n-2} W_k XNOR(\hat{u}_{ik}, \hat{v}_{ik}), \quad (6)$$

where $\hat{U}$ and $\hat{V}$ are the quantized input vectors, $S_{\hat{u}_i}$ and $S_{\hat{v}_i}$ are the signs, $\hat{u}_{ik}$ and $\hat{v}_{ik}$ are the bits of each element of the input vectors, W_k is the Hamming similarity weight, and n is the bit width of the fixed-point representation. The proposed vector similarity is measured by adding the Hamming similarity between the elements $\hat{u}_i$ and $\hat{v}_i$ of the fixed-point input vectors $\hat{U}$ and $\hat{V}$ as Equation 6.

The similarity value between two fixed-point scalars $\hat{u}_i$ and $\hat{v}_i$ should be large as the bits of the integer and fractional parts $\hat{u}_{ik}$ and $\hat{v}_{ik}$ are similar if the signs $S_{\hat{u}_i}$ and $S_{\hat{u}_i}$ of the two scalars are the same, and small if the signs are different. Further, in the case of different signs, the similarity value should be lower than when the signs are the same regardless of the similarity of the other bits. Thus, the signs of the two scalars are multiplied by the Hamming similarity of the bits of the integer and fractional parts as shown in Equation 6.

Each bit of the fixed-point representation indicates a different value depending on its position. Hence, for more accurate similarity measurement, the similarity value between two fixed-point scalars was measured with weight. We used the weight as $W_k = 2^{k+\alpha-n}$ for the vector similarity measure of Q-MANN without significantly increasing the computation-energy consumption. The weight constant α was empirically determined through various experiments.

For the end-to-end training using gradient back-propagation and SGD as a conventional MANN, all components of a MANN must be differentiable. Thus, we smoothed a quantization function for gradient back-propagation while we used it as a step-wise function in forward propagation. In addition, since the Hamming distance is identical to the Manhattan distance for binary vectors and the Manhattan distance is differentiable, the proposed function can be used in the end-to-end training by using SGD and gradient back-propagation. Furthermore, the computation-energy consumption of gradient back-propagation can be lowered through an approximation that uses only addition and shift operations. Consequently, the proposed gradient back-propagation function is defined as

$$\frac{\partial S_H(\hat{U}, \hat{V})}{\partial \hat{u}_i} \approx S_{\hat{u}_i} 2^{\alpha} (S_{\hat{u}_i} - S_{\hat{v}_i}) - \sum_{k=0}^{n-2} (S_{\hat{v}_i} 2^{\alpha} (\hat{u}_{ik} - \hat{v}_{ik})). \quad (7)$$

As shown in Figure 2A, Q-MANN did not expand the distribution of similarity even when training progressed. To investigate the effect of the quantization error on the distribution of similarity in Q-MANN, $FRAC$ was reduced to 7, 4, and 1 with IWL of 5 and the distribution was mea-

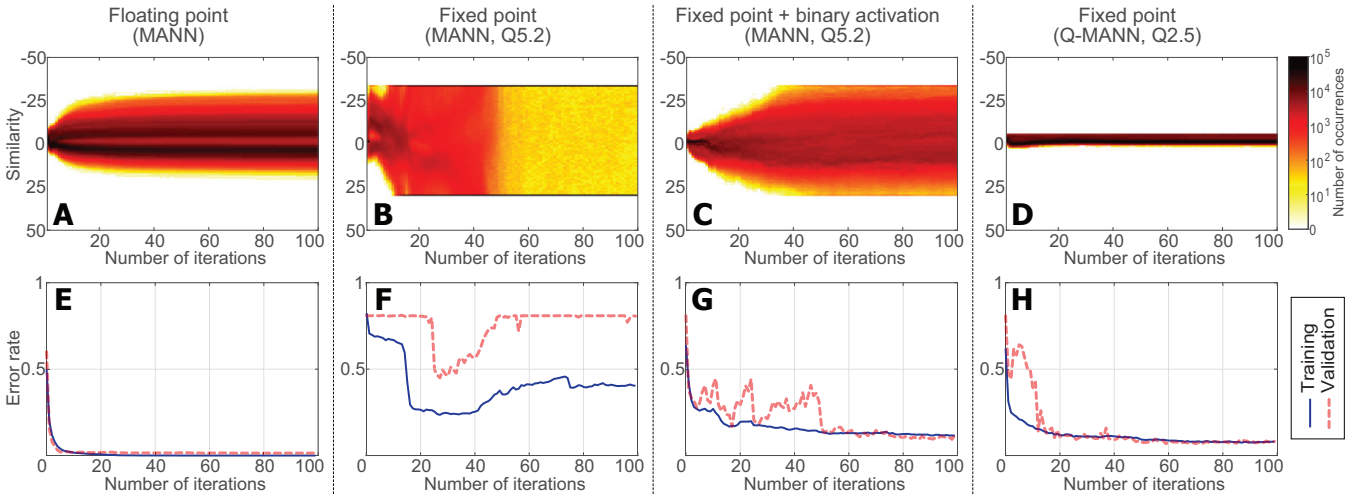

Figure 3: The results for the conventional MANN and Q-MANN on the bAbI dataset (task8), (A)-(D): the distribution of the similarity measure, (E)-(H): the error rate of training and validation. (A) and (E) were the results with a 32-bit floating-point representation. As depicted in (A), the distribution width of cosine similarity became wider as the training progressed. In addition, when applying a fixed-point quantization, the width became much wider than that of floating-point representation due to the quantization error as (B). Thus, this training tendency of cosine similarity caused the fixed-point overflow, which considerably degraded the learning performance when training with fixed-point parameters and activations as (F). By applying the binary quantization to activations, multiplications can be replaced by additions in cosine similarity, which could prevent numerous fixed-point overflows as (C). However, due to the lack of information in the similarity measure, the learning performance was limited as (G). Q-MANN using bounded similarity (Hamming similarity) was suitable for applying the fixed-point quantization with a limited numerical representation because the distribution width of the similarity did not increase as training progressed shown in (D). The result of Q-MANN (H) showed a lower error rate than those of the conventional MANN (F and G) when applying the 8-bit fixed-point quantization.

sured as in the experiments on the conventional MANN (Figures 2A, 2B, and 2C, respectively). The results showed that there was no significant difference in the width of the distributions. They were quite different from those in the experiments on the conventional MANN where the distribution width became wider as $FRAC$ decreased.

To avoid overflow in a fixed-point quantization with a limited numerical representation range, Q-MANN was optimized for a narrow distribution of vector similarity. Because of this, even if many bits were used, the accuracy would become poor for a significant gap between the actual width of the distribution and the numerical representation range. Thus, the use of optimized IWL (Q2.7, Figure 2H) improved the learning performance compared to the case of using more IWL (Q5.7, Figure 2E).

Since Q-MANN is trained with a narrow distribution of similarity, it can prevent the fixed-point overflow. This enhances the robustness for the quantization error and stabilizes training by applying a fixed-point quantization. In addition, the similarity measure function of Q-MANN uses only simple operations such as addition, shift, and XNOR gate operations. Hence, Q-MANN is suitable in a limited-resource environment.

Effective training techniques

One of the other reasons for the poor learning performance was in the RNN used as the memory controller when applying a fixed-point quantization to a MANN. The RNN uses the same parameters for different time steps. Thus, the influence of the error can be greatly magnified, because the quantization error of the parameter is equal for the time steps. We reduced the influence of the deterministic quantization error in the memory controller by employing slightly different IWL and $FRAC$ at every time step while maintaining the bit width used for quantization. We named this method *Memory controller Quantization control* (MQ). The use of MQ made it possible to vary the quantization error at every time step without increasing the quantization overhead, such that the error could be canceled each other.

In addition, as shown in Figure 3B, when the fixed-point overflow occurred in the similarity measure due to a fixed-point quantization, the training and validation error increased sharply (Figure 3F). With *Early Stopping* (ES), we could reduce the degradation and variance of learning performance caused by the quantization error.

Experimental Results and Discussion

We verified the performance of the proposed Q-MANN and training method by applying a fixed-point quantization. All experiments were performed on the bAbI dataset (10k) (Weston et al. 2015), by using a training method and hyperparameters similar to those in (Sukhbaatar et al. 2015). Details of the model and hyperparameters are provided in Supplementary Table 6 and Table 7.

Table 2: Test error rates (%) on the bAbI dataset for various configurations and their computation-energy gain (ES=Early Stopping, MQ=Memory controller Quantization control)

		bit width			error rate (%)		computation -energy
	type	input	parameter	activation	avg. of best	avg. of mean	gain[a]
MANN	floating	1	32	32	15.33	17.14	1×
MANN	fixed	1	8	8	40.04	51.23	20.22×
MANN + ES	fixed	1	8	8	39.31	43.87	20.22×
MANN + ES + MQ	fixed	1	8	8	33.67	38.50	20.22×
Q-MANN	fixed	1	8	8	24.33	30.43	21.03×
Q-MANN + ES	fixed	1	8	8	26.47	32.88	21.03×
Q-MANN + ES + MQ	fixed	1	8	8	**22.30**	**27.68**	21.03×
MANN	fixed	1	8	1	30.81	44.49	22.25×
MANN + ES	fixed	1	8	1	31.41	37.12	22.25×
MANN + ES + MQ	fixed	1	8	1	30.14	42.24	22.25×
Q-MANN	fixed	1	8	1	27.11	31.91	22.25×
Q-MANN + ES	fixed	1	8	1	26.66	32.66	22.25×
Q-MANN + ES + MQ	fixed	1	8	1	**25.63**	**31.00**	22.25×

[a] calculated by using data from (Horowitz 2014)

In a manner similar to that in (Sukhbaatar et al. 2015), we repeated each training 10 times to obtain the best and mean error rate of each task in the bAbI dataset. We used the average of the best and mean error rates for each task as the two types of performance measuring metrics.

The optimal fixed-point representation for IWL and $FRAC$ was obtained through experiments. The optimal learning performance of the 8-bit fixed-point quantization was obtained when using Q5.2 for a conventional MANN and Q2.5 for Q-MANN.

Table 2 provides the experimental results under various conditions. The entire results are included in Supplementary Table 4 and Table 5. The binary vector was used as the input in the form of Bag-Of-Words for the computation efficiency. The baseline configuration of our experiments adopted 32-bit floating-point parameters and activations with the binary input.

We obtained a gain of about 20× in the computational energy compared to the baseline when we applied 8-bit fixed-point quantization (Q5.2) to a conventional MANN. However, the average error rates of the best and mean cases increased by 160% and 200%, respectively. The learning performance significantly deteriorated (Figure 3B) because the quantization error greatly increased due to the fixed-point overflow in the similarity measure (Figure 3F). We achieved an increase of 159% in the average error rate of the mean case by applying ES. The error rate was increased by 120% by applying both ES and MQ compared to the baseline.

In the case of Q-MANN, the 8-bit fixed-point quantization (Q2.5) resulted in a gain of about 21× in the computational energy compared to the baseline and the average error rate of the best and mean cases were reduced by about 37% and 41%, respectively, compared with that of the conventional MANN. As shown in Figure 3D, these results can be attributed to the infrequent occurrence of the fixed-point overflow at the similarity measure in Q-MANN, hence it became robust to the quantization error as in Figure 3H. As a result of applying ES and MQ to Q-MANN, the error rate was reduced by up to 46% compared with the case of the conventional MANN.

We applied the binary quantization to activations of the conventional MANN with 8-bit fixed-point quantized parameters. As a result, we obtained a gain in the computational energy of about 22× compared to that of the baseline. In addition, the average error rate of the best and mean cases increased by about 100% and 160% compared to the baseline, which showed a lower increase in the error rate compared to that of the 8-bit fixed-point quantization. This was because overflows were reduced by the binarization as in Figure 3C. Hence, the quantization error caused by the fixed-point overflow had a greater impact on the learning performance of the conventional MANN than the lack of information due to the binarization. As a result of training with the binarization, the error rate of Q-MANN with ES and MQ could be reduced by up to 17% and 30%, respectively, compared to the conventional MANN because Q-MANN could use enough information while preventing the fixed-point overflow.

The ultimate goal of this study is to train a MANN in a limited-resource environment through a quantization, which allows us to use dot product instead of cosine similarity as an approximation to reduce the amount of computational energy consumption. Our analysis showed that a conventional MANN with dot product is vulnerable to the quantization error. Since cosine similarity is a normalized value of dot product, it implies that a MANN with cosine similarity is also vulnerable to the quantization error. Hence, the advantage of Q-MANN proposed in this paper is still valid using cosine similarity.

Conclusion

In this paper, we applied fixed-point and binary quantization to a conventional MANN in both of training and inference as a pioneering study. Through theoretical analysis and various experiments, we revealed that the quantization error had a great impact on the learning performance of a conventional

MANN. Based on our in-depth analysis, we proposed Q-MANN which use bounded similarity in content-based addressing, which is suitable for fixed-point and binary quantization with a limited numerical representation. We also showed that Q-MANN could be converged and achieve more robust learning performance in comparison to a conventional MANN for fixed-point and binary quantization.

Acknowledgements

This research was supported by Ministry of Science, ICT and Future Planning (Basic Science Research Program [2016M3A7B4911115] and Project for Research and Development of Police science and Technology [PA-C000001]) and Samsung Research Funding Center of Samsung Electronics [SRFC-IT1601-05].

References

Allison, G.; Chris, G.; Ryan, O.; and Shashank, P. 2017. Deploying deep neural networks with nvidia tensorrt. https://devblogs.nvidia.com/parallelforall/deploying-deep-learning-nvidia-tensorrt/.

Courbariaux, M.; Bengio, Y.; and David, J.-P. 2014. Training deep neural networks with low precision multiplications. *arXiv preprint arXiv:1412.7024*.

Courbariaux, M.; Bengio, Y.; and David, J.-P. 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems*, 3123–3131.

Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*.

Graves, A.; Wayne, G.; Reynolds, M.; Harley, T.; Danihelka, I.; Grabska-Barwińska, A.; Colmenarejo, S. G.; Grefenstette, E.; Ramalho, T.; Agapiou, J.; et al. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature* 538(7626):471–476.

Graves, A.; Wayne, G.; and Danihelka, I. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Gupta, S.; Agrawal, A.; Gopalakrishnan, K.; and Narayanan, P. 2015. Deep learning with limited numerical precision. In *ICML*, 1737–1746.

Horowitz, M. 2014. 1.1 computing's energy problem (and what we can do about it). In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*, 10–14. IEEE.

Hubara, I.; Courbariaux, M.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Quantized neural networks: Training neural networks with low precision weights and activations. *arXiv preprint arXiv:1609.07061*.

Lin, D. D., and Talathi, S. S. 2016. Overcoming challenges in fixed point training of deep convolutional networks. *arXiv preprint arXiv:1607.02241*.

Lin, D. D.; Talathi, S. S.; and Annapureddy, V. S. 2015. Fixed point quantization of deep convolutional networks. *arXiv, page*.

Ott, J.; Lin, Z.; Zhang, Y.; Liu, S.-C.; and Bengio, Y. 2016. Recurrent neural networks with limited numerical precision. *arXiv preprint arXiv:1608.06902*.

Rastegari, M.; Ordonez, V.; Redmon, J.; and Farhadi, A. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, 525–542. Springer.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Sukhbaatar, S.; Weston, J.; Fergus, R.; et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, 2440–2448.

Tang, W.; Hua, G.; and Wang, L. 2017. How to train a compact binary neural network with high accuracy? In *Thirty-First AAAI Conference on Artificial Intelligence*.

Weston, J.; Bordes, A.; Chopra, S.; Rush, A. M.; van Merriënboer, B.; Joulin, A.; and Mikolov, T. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Weston, J.; Chopra, S.; and Bordes, A. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.

Zhou, S.; Wu, Y.; Ni, Z.; Zhou, X.; Wen, H.; and Zou, Y. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*.