Interspeech 2022
18-22 September 2022, Incheon, Korea

Towards Green ASR: Lossless 4-bit Quantization of a Hybrid TDNN System on the 300-hr Switchboard Corpus

Junhao Xu, Shoukang Hu, Xunying Liu, Helen Meng

The Chinese University of Hong Kong, Hong Kong SAR, China

jhxu, skhu, xyliu, hmmeng@se.cuhk.edu.hk

Abstract

State-of-the-art time automatic speech recognition (ASR) systems are becoming increasingly complex and expensive for practical applications. This paper presents the development of a high performance and low-footprint 4-bit quantized LF-MMI trained factored time delay neural networks (TDNNs) based ASR system on the 300-hr Switchboard corpus. A key feature of the overall system design is to account for the fine-grained, varying performance sensitivity at different model components to quantization errors. To this end, a set of neural architectural compression and mixed precision quantization approaches were used to facilitate hidden layer level auto-configuration of optimal factored TDNN weight matrix subspace dimensionality and quantization bit-widths. The proposed techniques were also used to produce 2-bit mixed precision quantized Transformer language models. Experiments conducted on the Switchboard data suggest that the proposed neural architectural compression and mixed precision quantization techniques consistently outperform the uniform precision quantised baseline systems of comparable bit-widths in terms of word error rate (WER). An overall "lossless" compression ratio of 13.6 was obtained over the baseline full precision system including both the TDNN and Transformer components while incurring no statistically significant WER increase.

Index Terms: TDNN, ASR, Neural Architecture Compression, Neural Network Quantization, Mixed Precision

1. Introduction

State-of-the-art automatic speech recognition (ASR) systems represented by both the traditional hybrid HMM-DNN architectures [1, 2, 3, 4, 5, 6, 7, 8, 9] and the recently emerging all neural end-to-end (E2E) modelling paradigm [10, 11, 12, 13, 14, 15] are becoming increasingly complex and expensive for practical applications. For example, the best E2E encoder-decoder ASR system presented in [16] contained up to 280.1 million model parameters. This not only lead to a large increase in their computational cost when operating on the cloud, but also creates difficulty when deploying them on edge devices to preserve privacy and reduce latency.

An ultimate goal for many data and resource intensive deep learning based AI applications, including ASR systems, is to derive "lossless" model compression approaches that allow high performance and low-footprint speech recognition systems to be constructed while incurring minimum performance degradation. To this end, one efficient solution is to use low-bit deep neural network (DNN) quantization techniques [17, 18, 19, 20], which has drawn increasing interest in the machine learning and speech technology community in recent years. By replacing floating point weights with low precision values, the resulting quantization methods can significantly reduce the model size and inference time without modifying the model architectures. Traditional DNN quantization approaches [21, 22, 23] are predominantly based on uniform precision, where a manually defined identical bit-width is applied to all weight parameters. This fails to account for the varying performance sensitivity at different parts of the system to quantization errors.

This paper presents the development of a high performance and ultra-compact 4-bit quantized LF-MMI trained factored time delay neural networks (TDNNs) based ASR system on the 300-hr Switchboard corpus. A key feature of the overall system design is to account for the fine-grained, varying performance sensitivity at different model components to compression and quantization errors. This allows the trade-off between model compression ratio and accuracy performance target. The overall system development contained two stages. In first stage, DARTS [24] based neural architectural compression approaches tailor designed for TDNNs [15] are utilized first to minimize the model structural redundancy in full precision models first to facilitate hidden layer level auto-configuration of optimal factored TDNN weight matrix subspace dimensionality. This severs as a streamlined starting point for the following mixed precision quantization stage, where minimum KL divergence, log-likelihood curvature or mixed precision architectural search based approaches [25, 26, 27] are adopted to learn the optimal TDNN layer level quantization bit-widths. To overcome the difficulty in using gradient descent methods to directly estimate discrete quantized TDNN model parameters, alternating direction methods of multipliers (ADMM) [22] and quantization aware training (QAT) [28] are also used in the pre-training and fine-tuning stages before and after the locally varying precisions are determined. The proposed techniques were also used to produce layer level 2-bit mixed precision quantized Transformer language models [25, 26].

Experiments conducted on the 300-hr benchmark Switchboard data suggest that the proposed neural architectural compression and mixed precision quantization techniques consistently outperform the uniform precision quantised baseline TDNN systems of comparable bit-widths in terms of word error rate (WER) reductions up to 1.5% absolute. An overall "lossless" compression ratio of 13.6 was obtained over the TDNN and Transformer models while incurring no statistically significant WER increase over the baseline 32-bit full precision system. In addition, if adopting a more generous 8-bit mixed precision based TDNN quantization, corresponding to an overall system compression ratio of 9.3, a marginal WER reduction was obtained over the full precision system.

The main contributions of this paper are summarized as follows. First, this paper presents the first application of neural architectural compression and mixed precision quantization techniques to state-of-the-art LF-MMI trained hybrid TDNN and Transformer LM based ASR systems that already benefit from a compact model structure design using low-rank weight matrix factorization [1]. Latest research [29] combines neural archi-

Copyright © 2022 ISCA

10.21437/Interspeech.2022-678

tecture search and quantization to key word spotting tasks, but manual defined precision setting is applied to weight and activation parameters. Similar approaches using uniform or manually configured quantization precision were previously investigated only in the context of hybrid BLSTM and RNN transducer based systems [23, 28]. Second, a largest "lossless" system compression ratio of 13.6 was obtained over the full precision 32-bit baseline without statistically significant WER increase on the benchmark 300-hr Switchboard data. This is in contrast to prior researchers using low-bit quantization approaches on the same task where statistically significant accuracy performance degradation was reported [23].

The rest of this paper is organized as follows. TDNN system with low-rank weight factorization is reviewed in Section 2. Neural architecture search based TDNN model compression approaches are presented in Section 3. Uniform precision neural network quantization methods are reviewed in Section 4. Section 5 presents three mixed precision quantization methods. Experiments and results are shown in Section 6. Finally, conclusions and future work are discussed in Section 7.

2. Time Delay Neural Network

Time delay neural networks (TDNNs) [30, 31] based hybrid HMM-DNN acoustic models in recent years defined state-of-the-art speech recognition performance over a wide range of tasks. In particular, the recently proposed lattice-free MMI trained factored TDNN systems [14] benefits from a compact model structure featuring low-rank weight matrix factorization while remain highly competitive performance wise against end-to-end approaches to date.

TDNNs can be considered as a special form of one-dimensional convolutional neural networks (CNNs) [32] when parameters are tied across different time steps. The bottom layers of TDNNs are designed to learn a narrower temporal context span, while the higher layers to learn wider, longer range temporal contexts. To further reduce the risk of overfitting to limited training data and the number of parameters, a factored TDNN (TDNN-F) model structure was proposed in [1], which compresses the weight matrix by using semi-orthogonal low-rank matrix factorization. In the TDNN-F model, the hidden layer specific bottleneck projection dimensionality settings present the group of hyper-parameter that needs to be determined in this paper using neural architecture compression prior to layer level low precision quantization is applied.

2.1. Neural Architecture Search

In this section, we introduce the pipelined Gumbel-softmax differentiable neural architecture search (DARTS) methods adopted in this paper. The general form of DARTS architecture selection methods [24, 33]are introduced as follows. For example, the l-th neural network hidden layer output $\boldsymbol{h}^l$ can be computed as a linear combination between the architecture weights λ_i^l and candidate architecture choices $\phi_i^l(\cdot)$ in the DARTS super-network:

$$\boldsymbol{h}^l = \sum_{i=0}^{N^l-1} \lambda_i^l \phi_i^l(\boldsymbol{W}_i^l \boldsymbol{h}^{l-1}) \tag{1}$$

where λ_i^l is the architecture weight for the i-th candidate choice in the l-th layer, N_l is the total number of choices in this layer. The precise form of neural architectures being considered at this layer is determined by the linear transformation $\boldsymbol{W}_i^l$ and activation function $\phi(\cdot)$ used by each candidate system. For example, when selecting the dimensionality of the bottleneck linear projection at each hidden layer, the linear transformation $\boldsymbol{W}_i^l = \tilde{\boldsymbol{W}}_i^l \tilde{\boldsymbol{W}}_i^{lT}$ is a decomposed matrix of the i^{th} projection dimensionality, while $\phi(\cdot)$ is also an identity matrix.

2.2. Gumbel-softmax DARTS

For traditional DARTS methods, when similar architecture weights are obtained using a flattened Softmax function, the confusion over different candidate systems increases and search errors may occur. To this end, a Gumbel-Softmax distribution [33, 34, 35] is used to sharpen the architecture weights to produce approximately a one-hot vector. This allows the confusion between different architectures to be minimised. The architecture weights are computed as,

$$\lambda_i^l = \frac{exp((log\gamma_i^l + G_i^l)/T)}{\sum_{j=0}^{N^l-1} exp((log\gamma_j^l + G_j^l)/T)} \tag{2}$$

where $G_i^l = -log(-log(U_i^l))$ is the Gumbel variable, and U_i^l is a uniform random variable. When the temperature parameter T approaches 0, it has been shown that the Gumbel-Softmax distribution is close to a categorical distribution [35]. Different samples of the uniform random variable U_i^l lead to different values of λ_i^l in Eq. 1. The loss function gradient w.r.t $log\gamma_k^l$ is computed as an average over J samples of the architecture weights,

2.3. Pipelined DARTS

For the above Gumbel-Softmax DARTS systems, the search algorithms may prematurely select sub-optimal architectures at an early stage. Inspired by [36], we decouple the update of normal DNN parameters and architecture weights into two separate stages performed in sequence. This leads to the pipelined DARTS approach. In order to prevent overfitting to the training data, a separate held-out data set taken out of the original training data is used. we randomly select 5% of the original training set as the held-out data set. In Pipelined DARTS systems, the normal DNN parameters are updated to convergence on the training data first (not containing the separate held-out data), while randomly sampled one-hot architecture weights drawn from a uniform distribution are used. In the following stage, we fix the normal DNN parameters estimated in the first stage in the super-network and update the architecture weights using the held-out data for the Gumbel-Softmax DARTS. This produces the Pipelined Gumbel-softmax DARTS (PipeGumbel) systems. To further adjust the trade-off between system performance and compression ratio, a penalized loss function incorporating the underlined neural network size is used, given as follows:

$$L = L_{LF-MMI} + \eta \sum_{l,i} \lambda_i^l C_i^l \tag{3}$$

where L_{LF-MMI} is the lattice-free MMI criterion. C_i^l is the complexity penalty term expressed as the model parameters of the i-th candidate considered at the l-th layer. η is the penalty scaling factor empirically set for different tasks.

3. Neural Network Quantization

For a standard n-bit quantization problem of neural networks, we consider a full precision weight parameter Θ and find its closest discrete approximation from the following quantization table $Q \in \{0, \pm 1, \pm 2, \ldots, \pm(2^{n-1}-1)\}$ as

$$f(\Theta) = \arg\min_{Q} |\Theta - Q| \tag{4}$$

while one bit is reserved to denote the sign. With further simplification, low bit quantization, for example, binarization $\{1, -1\}$ [37, 19] and ternary $\{-1, 0, 1\}$ [38], can be produced.

Table 1: Performance (WER%, #parameters) comparison of TDNN-F models configured with projection dimensionality of the Kaldi recipe baseline system, and the pipelined Gumbel-softmax DARTS (PipeGumbel) system. η is the penalty factor. The dimensionality choices: {25,50,80,100,120,160,200,240}. * denotes a statistically significant WER difference over the baseline system (sys. 1).

ID	Method	η	Bottleneck Dim Index	eval2000		rt02			rt03		WER(%)	#param	Develop
				swbd	callhm	swbd1	swbd2	swbd3	fsh.	swbd	avg.		Time
1	Baseline	-	5 5 5 5 5 5 5 5 5 5 5 5 5 5 5	9.7	18.0	11.5	15.3	20.0	12.6	19.5	15.5	18.6M	30h
2	PipeGumbel	0.1	2 3 2 0 2 2 2 1 2 1 0 3 5 5	9.2*	17.4*	10.9*	14.4*	19.5*	11.9*	19.2*	15.0*	12.4M	26h

When applying quantization to all weight matrices in the model, we can use a more general format in equation (5) to represent the quantization for each parameter. Let $\Theta_i^{(l)}$ be the i^{th} parameter within any of the l^{th} weight cluster, for example, all weight parameters of the same TDNN hidden layer,

$$f(\Theta_i^{(l)}) = \arg\min_{Q_i^{(l)}} |\Theta_i^{(l)} - Q_i^{(l)}| \tag{5}$$

The locally shared l^{th} quantization table is given by

$$Q_i^{(l)} \in \{0, \pm\alpha^{(l)}, \ldots, \pm\alpha^{(l)}(2^{n_l-1}-1)\} \tag{6}$$

where $\alpha^{(l)}$ is a full precision scaling factor used to adjust the dynamic range of all the unquantized weights in the cluster. It is shared locally among weight parameters clusters. A special case, when the local quantization table in equation (5) is shared across all the layers, leads to the traditional uniform precision quantization approach. The only remaining factor affecting the system performance is the bit length n_l which is also globally set to be $1, 2, 4, 8, 16$ etc. Efficient alternating direction methods of multipliers (ADMM) [22] based optimization of local quantization tables can be used.

4. Mixed Precision Quantization

This section presents three approaches to automatically learn the optimal local precision settings for TDNN-F systems.

4.1. KL Divergence Based Mixed Precision Quantization

In order to minimize the distance between the distribution embodied by the full precision system and that of the quantized model, Kullback-Leibler (KL) divergence between full precision and quantized NNs is used to measure the resulting performance sensitivity. Taking a L-layer TDNN-F for example, for any quantization $f(\cdot)$ being applied to the full precision parameters $\boldsymbol{\Theta}$, the KL divergence based quantization sensitivity measure is computed over the input spectrum of T frames:

$$\begin{aligned}\Omega^{\mathrm{KL}} &= \sum_{i=1}^{L} \Omega_i^{\mathrm{KL}} = \sum_{i=1}^{L} D_{\mathrm{KL}}(P(\boldsymbol{\Theta_i})||P(f_{n_i}(\boldsymbol{\Theta_i}))) \\ &= \sum_{i=1}^{L}\sum_{t=1}^{T} P(\sigma(\boldsymbol{h}_l^t), \boldsymbol{\Theta}_i) \ln \frac{P(\sigma(\boldsymbol{h}_l^t), \boldsymbol{\Theta}_i)}{P(\sigma(\boldsymbol{h}_l^t), f_{n_i}(\boldsymbol{\Theta}_i))}\end{aligned} \tag{7}$$

where $\boldsymbol{\Theta}_i$ denote the full precision parameters of the i^{th} layer, and $f_{n_i}(\boldsymbol{\Theta}_i)$ is n_i-bit quantized parameters given a particular local precision bit width n_i and $\boldsymbol{h}_t^l$ is the TDNN-F output vector computed at layer l. When computing the KL metric in Eqn. (7), $\boldsymbol{h}_t^l$ is fed into a Sigmoid gate first to produce normalised, probability like outputs between 0 and 1, $P(\sigma(\boldsymbol{h}_t^l), \boldsymbol{\Theta})$. Given a target model size constraint (e.g. average 4-bit precision), the KL metric for each precision setting of each layer is computed and minimized to select the optimal local bit-width while satisfying the constraint[1].

[1]Based on the performance sensitivity ranking measured by either the KL metric of Eqn. (7), or the curvature metric of Eqn. (8), the optimal local bit-widths combination that is closest to the target average quantization precision, e.g. 4-bit, while producing the minimum KL or curvature measured performance sensitivity, will be selected.

4.2. Curvature Based Mixed Precision Quantization

The second approach minimizes the performance sensitivity to quantization by examining the local training data loss function curvature. Under mild assumptions such that the parameters of a DNN is twice differentiable and while converging to a local optimum, it is shown in [39, 40] that the separation performance sensitivity to quantization, when using a given precision setting, can be expressed as the squared quantization error further weighted by the parameter Hessian matrix trace. For any quantization $f(\cdot)$ being applied to the parameters $\boldsymbol{\Theta}$ of the L-layer TDNN-F acoustic model, the total performance sensitivity is given by the sum of Hessian trace weighted squared quantization error, to be minimized under a target model size constraint.

$$\Omega^{\mathrm{Hes}} = \sum_{i=1}^{L} \Omega_i^{\mathrm{Hes}} = \sum_{i=1}^{L} Tr(\boldsymbol{H}_i) \cdot ||f(\boldsymbol{\Theta}_i) - \boldsymbol{\Theta}_i||_2^2 \tag{8}$$

An efficient Hessian-free approach based on the Huchinson's Algorithm [40, 41] is used to approximate the Hessian trace.

4.3. Architecture Search Based Mixed Precision Quantization

The third solution to automatically learn the optimal local quantization precision settings is to use mixed precision based neural architecture search (NAS) [42, 43] approaches. The supernetwork is constructed by first using ADMM to separately pretrain each TDNN-F layers using different uniform precision settings, before treating these uniform precision as different NAS candidate at each layer. In order to avoid the trivial selection of the longest, most generous quantization bit width, these precision selection weights learning can be further constrained by a model complexity penalty term with respect to the number of bits retained after quantization, in order to obtain a target average quantization precision, for example, 4-bit, akin to the penalized LF-MMI criterion in Eq. (3).

5. Experiments

The baseline full precision Kaldi recipe[2] TDNN-F system (line 0, Table 1 and 2) was trained on the 300-hour Switchboard corpus featuring speed perturbation and i-Vector speaker adaptation. Model training and evaluation time measurement were conducted with one thread on a single NVIDIA Tesla V100 Volta GPU. Matched pairs sentence-segment word error (MAPSSWE) based statistical significance test was performed at a significance level $\alpha = 0.05$.

In the first stage of system development, the penalized Pipelined Gumbel-softmax DARTS (*PipeGumbel*) approach of Section 2.3 was used to compress baseline TDNN-F system's hidden layer projection dimensions from a uniform set recipe configuration of 200 down to as low as 25. The resulting *PipeGumbel* compressed TDNN-F system (line 2, Table 1 and 2) produced significant WER reductions of 0.5% averaged

[2]All of this is in published Kaldi code at https://github.com/kaldi-asr/kaldi/tree/master/egs/swbd/s5c/run.sh and https://github.com/kaldi-asr/kaldi/tree/master/egs/swbd/s5c/local/chain/tuning/run_tdnn_7q.sh.

Table 2: Performance of the baseline full precision, uniform precision and mixed precision quantized TDNN-F systems with local precision settings automatically learned using the HES/KL/NAS methods introduced in Section 4 on Switchboard NIST Hub5'00, RT02 and RT03. RNNLM rescoring also applied. * and † denote no statistically significant WER difference obtained over the two full precision systems (sys.1, sys.0) respectively. Speech processing real time factors (RTFs) in the last column.

ID	quant. prec.	param. estim.	quant. method	#bit	Hub5'00		rt03		rt02			WER(%)	model size(MB)	comp. ratio	RTF $*10^{-3}$
					swbd	callhm	fsh.	swbd	swbd1	swbd2	swbd3	avg.			
0	Baseline			32	9.7	18.0	12.6	19.5	11.5	15.3	20.0	15.5	74.4	-	0.223
1	PipeGumbel			32	9.2	17.4	11.9	19.2	10.9	14.4	19.5	15.0	49.6	1.5	0.187
2	*uniform* prec.	BP [21] *(modified)*	Offline Tuning	1	16.4	25.0	18.3	27.2	17.6	20.9	27.7	22.3	1.9	39.2	0.065
3				2	15.4	23.9	17.2	26.5	16.8	20.1	26.9	21.7	3.7	20.1	0.103
4				4	11.0	20.1	13.6	22.5	13.0	16.2	22.7	16.9	6.4	11.6	0.105
5				8	10.3	18.7	12.6	20.4	11.8	15.2	20.6	15.8†	12.6	5.9	0.097
6				16	9.0	17.2	11.6	19.1	10.8	14.2	19.2	14.8	25.1	3.0	0.111
7		QAT [28]	Offline Quant	1	15.0	23.1	16.2	25.0	15.5	18.8	25.1	19.7	1.9	39.2	0.057
8				4	10.6	19.1	13.0	21.7	12.4	15.6	21.5	16.1	6.4	11.6	0.090
9				8	9.7	18.0	12.2	19.8	11.2	14.9	19.9	15.4†	12.6	5.9	0.096
10				16	9.2	17.5	11.8	19.3	11.0	14.3	19.6	15.0	25.1	3.0	0.120
11	*mixed* prec.	BP [21] *(modified)*	Hes	**4**	**10.2**	**18.6**	**12.4**	**19.8**	**11.5**	**14.9**	**20.0**	**15.6**†	**6.3**	**11.8**	0.073
12				8	9.5	17.5	12.1	19.5	11.1	14.8	19.6	15.2*	12.6	5.9	0.077
13		QAT [28]		4	9.7	17.7	12.3	19.8	11.3	15.0	19.7	15.5†	6.4	11.6	0.071
14				8	9.3	17.5	12.1	19.5	11.2	14.7	19.5	15.2*	12.5	6.0	0.080
15		BP [21] *(modified)*	KL	4	10.3	18.5	12.4	19.9	11.4	15.0	20.1	15.6†	6.3	11.7	0.074
16				8	9.4	17.6	12.1	19.4	11.0	14.8	19.7	15.2*	12.6	5.9	0.079
17		QAT [28]		4	9.8	17.8	12.2	19.7	11.2	14.9	19.8	15.4†	6.4	11.6	0.069
18				**8**	**9.3**	**17.6**	**12.0**	**19.4**	**11.0**	**14.6**	**19.6**	**15.1***	**12.5**	**5.9**	0.081
19		BP [21] *(modified)*	NAS *gumbel*	4	10.5	18.8	12.8	20.6	11.9	15.5	21.0	16.0	6.3	11.8	0.072
20				8	9.8	18.0	12.2	19.7	11.1	14.7	19.9	15.4†	12.6	5.9	0.078

Table 3: Performance of best 4-bit and 8-bit quantized TDNN-F systems after 2-bit quantized Transformer LM rescoring.

ID	sys.	AM	LM	Hub5'00 swbd/chm	rt03 fsh/swbd	rt02 swb1/swb2/swb3	model size(MB)
1	Baseline (sys. 0, Tab 2)	32b	32b	9.6/17.8	12.4/19.3	11.3/15.1/19.9	180.4
2	PipeGumbel (sys. 1, Tab 2)	32b	32b	9.0/17.3	11.7/19.1	10.8/14.2/19.3	155.6
3	Baseline+LHUC	32b	32b	7.7/15.1	9.8/16.0	9.4/12.2/15.8	180.4
4	PipeGumbel+LHUC	32b	32b	7.5/14.4	9.6/15.6	9.0/11.8/15.6	155.6
5	TDNN-4bit (sys. 11, Tab 2)	4b	2b	10.0/18.3	12.2/19.5	11.4/14.8/19.6	13.3
6	TDNN-8bit (sys. 18, Tab 2)	8b	2b	9.2/17.3	11.7/19.2	10.8/14.4/19.4	19.5
7	TDNN-4bit+LHUC	4b	2b	8.5/15.6	10.3/16.9	10.0/12.8/17.1	13.3
8	TDNN-8bit+LHUC	8b	2b	7.7/14.8	9.9/16.0	9.4/12.3/16.0	19.5
9	IBM-BLSTM-300HR [23]	4b	-	10.4/18.3	-	-	-
10	IBM-BLSTM-2000HR [23]	4b	-	8.0/14.0	-	-	-
11	IBM-RNN-T-2000HR [23]	32/4b	-	8.8/17.4	-	-	-

across all three test sets while reducing the number of model parameters by 33% relative.

This system serves as the streamlined starting point of a range of quantized systems shown in Table 2 (line 2 to 20). Among these, several trends can be found. First, given the same quantization precision, for example, 4-bit, all the mixed precision quantized models introduced in Section 5, including curvature based HES (sys. 11, 13), KL (sys. 15, 17) and mixed precision NAS (sys. 19) outperform the 4-bit uniform quantized model (sys. 4). Second, quantization aware training (QAT) [28] consistently outperformed the modified back-propagation (BP) [21] [3] for both uniform or mixed precision systems. Third, among all the mixed precision quantization methods, the lowest WER was obtained using the 8-bit KL+QAT quantized model (sys. 18[4]), producing a compression ratio of 5.9 and 0.4% absolute WER reduction over the baseline (sys. 0). Lastly, the largest "lossless" compression ratio of 11.8 was obtained using the 4-bit HES+QAT quantized model (sys. 11), while incurring no statistically significant WER increase over the baseline full precision model (sys. 0).

In Table 3, the recognition outputs of the above two best performing 4-bit and 8-bit quantized systems (sys. 11, 18) were then rescored using a 2-bit quantized Transformer LM derived using a KL learned mixed precision method [25, 26] akin to previous results (line 5, 6 in Table 3), and further speaker adapted using learning hidden unit contribution (LHUC) [44] (line 7, 8). The performance of the comparable baseline and NAS compressed full precision systems (line 1, 2) are also shown. These serve to contrast with the recently published low-bit quantized ASR systems [23] reported in the literature (line 9, 10) on the same task. The combination of the 4-bit TDNN-F acoustic and 2-bit Transformer language models (line 5) produced an overall "lossless" compression ratio of 13.6 over the full precision baseline (line 1) and comparable WER. With the use of compact LHUC transforms of only 2000 parameters per speaker, speaker adaptation further reduced the WERs of the quantized TDNN-F systems by 1.5%-2.7% absolute on the NIST **Hub5'00** test data (line 7-8 vs. 5-6). All our quantized TDNN-F systems (line 5-8) outperformed the *IBM-BLSTM* system using the 300-hr SWBD data (line 9). The WERs of our *TDNN-8bit+LHUC* system (line 8) are comparable to those of the 2000-hr SWBD+Fisher data trained *IBM-BLSTM* system (line 10).

6. Conclusions

This paper presents the development of a high performance and low-footprint 4-bit quantized LF-MMI TDNN based ASR system on the 300-hr Switchboard corpus. Fine-grained performance sensitivity at different model components to quantization errors was accounted for using a combination of neural architectural compression and mixed precision quantization approaches, which were also used to produce low-bit quantized Transformer LMs. An overall "lossless" compression ratio of 13.6 was obtained over the baseline full precision system while incurring no statistically significant WER increase. Future researches focus on improving hardware implementation and tighter integration with speech enhancement front-ends.

7. Acknowledgement

This research is supported by Hong Kong RGC GRF grant No. 14200021, 14200218, 14200220, Innovation & Technology Fund grant No. ITS/254/19 and ITS/218/21.

[3] Low precision quantized parameters were first used in the forward pass to compute the error loss before full precision parameters are used in the backward pass to propagate the gradients for model update.

[4] The automatically learned TDNN-F hidden layer specific bit widths of this system are {16,8,8,1,4,4,4,2,4,2,1,4,8,8,16}.

8. References

[1] D. Povey *et al.*, "Semi-orthogonal low-rank matrix factorization for deep neural networks," *INTERSPEECH*, 2018.

[2] O. Abdel-Hamid *et al.*, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," *ICASSP*, 2012.

[3] ——, "Exploring convolutional neural network structures and optimization techniques for speech recognition," *INTERSPEECH*, 2013.

[4] A. Graves *et al.*, "Speech recognition with deep recurrent neural networks," *ICASSP*, 2013.

[5] A. Amberkar *et al.*, "Speech recognition using recurrent neural networks," *International Conference on Current Trends towards Converging Technologies*, 2018.

[6] V. Peddinti *et al.*, "A time delay neural network architecture for efficient modeling of long temporal contexts," *INTERSPEECH*, 2015.

[7] D. Povey *et al.*, "Purely sequence-trained neural networks for asr based on lattice-free mmi," *INTERSPEECH*, 2016.

[8] S. Hochreiter *et al.*, "Long short-term memory," *Neural computation*, 1997.

[9] H. Sak *et al.*, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *INTERSPEECH*, 2014.

[10] W. Chan *et al.*, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," *ICASSP*, 2016.

[11] A. Graves *et al.*, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," *ICML*, 2006.

[12] A. Vaswani *et al.*, "Attention is all you need," *NIPS*, 2017.

[13] L. Dong *et al.*, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," *ICASSP*, 2018.

[14] A. Gulati *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *INTERSPEECH*, 2020.

[15] S. Hu *et al.*, "Neural architecture search for lf-mmi trained time delay neural networks," *ICASSP*, 2021.

[16] Z. Tüske *et al.*, "Single headed attention based sequence-to-sequence model for state-of-the-art results on switchboard," in *Interspeech*, 2020.

[17] K. Yu *et al.*, "Neural network language model compression with product quantization and soft binarization," *TASLP*, 2020.

[18] Y. Qian and X. X, "Binary neural networks for speech recognition," *Frontiers of Information Technology & Electronic Engineering*, 2019.

[19] C. Leng *et al.*, "Extremely low bit neural network: Squeeze the last bit out with admm," in *AAAI*, 2018.

[20] R. Ma *et al.*, "Highly efficient neural network language model compression using soft binarization training," in *ASRU*, 2019.

[21] M. Courbariaux *et al.*, "Binaryconnect: training deep neural networks with binary weights during propagations," in *NIPS*, 2015.

[22] J. Xu *et al.*, "Low-bit quantization of recurrent neural network language models using alternating direction methods of multipliers," in *ICASSP*, 2020.

[23] A. Fasoli *et al.*, "4-Bit Quantization of LSTM-Based Speech Recognition Models," in *INTERSPEECH*, 2021.

[24] H. Liu *et al.*, "Darts: Differentiable architecture search," *ICLR*, 2019.

[25] J. Xu *et al.*, "Mixed precision low-bit quantization of neural network language models for speech recognition," *TASLP*, 2021.

[26] ——, "Mixed precision quantization of transformer language models for speech recognition," in *ICASSP*, 2021.

[27] ——, "Mixed precision dnn qunatization for overlapped speech separation and recognition," *ICASSP*, 2022.

[28] H. D. Nguyen *et al.*, "Quantization aware training with absolute-cosine regularization for automatic speech recognition." in *Interspeech*, 2020.

[29] P. David *et al.*, "End-to-end keyword spotting using neural architecture search and quantization," *ICASSP*, 2022.

[30] A. Waibel *et al.*, "Phoneme recognition using time-delay neural networks," *IEEE transactions on acoustics, speech, and signal processing*, 1989.

[31] A. Waibel, "Consonant recognition by modular construction of large phonemic time-delay neural networks," *Advances in neural information processing systems*, 1988.

[32] Y. LeCun *et al.*, "Convolutional networks for images, speech, and time series," *Handbook of Brain Theory and Neural Networks*, 1995.

[33] X. Dong *et al.*, "Searching for a robust neural architecture in four gpu hours," *CVPR*, 2019.

[34] S. Xie *et al.*, "Snas: stochastic neural architec- ture search," *ICLR*, 2019.

[35] C. J. Maddison *et al.*, "The concrete distribution: A continuous relaxation of discrete random variables," *ICLR*, 2017.

[36] Z. Guo *et al.*, "Single path one-shot neural architecture search with uniform sampling," *ECCV*, 2020.

[37] M. Rastegari *et al.*, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *ECCV*, 2016.

[38] F. Li *et al.*, "Ternary weight networks," *arXiv preprint arXiv:1605.04711*, 2016.

[39] Z. Dong *et al.*, "Hawq-v2: Hessian aware trace-weighted quantization of neural networks," *NeurIPS*, 2019.

[40] ——, "Hawq: Hessian aware quantization of neural networks with mixed-precision," in *ICCV*, 2019.

[41] H. Avron and T. S, "Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix," *JACM*, 2011.

[42] T. Elsken *et al.*, "Neural architecture search: A survey." *J. Mach. Learn. Res.*, 2019.

[43] S. Hu *et al.*, "Dsnas: Direct neural architecture search without parameter retraining," in *CVPR*, 2020.

[44] P. wietojanski *et al.*, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *TASLP*, 2016.