IMPACT OF LOW-PRECISION DEEP REGRESSION NETWORKS ON SINGLE-CHANNEL SOURCE SEPARATION

Enea Ceolini *Shih-Chii Liu*

Institute of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland
enea.ceolini@ini.uzh.ch, shih@ini.ethz.ch

ABSTRACT

Recent work on developing training methods for reduced precision Deep Convolutional Networks show that these networks can perform with similar accuracy to full precision networks when tested on a classification task. Reduced precision networks decrease the demand on the memory and computational power capabilities of the computing platform. This paper investigates the impact of reduced precision deep Recurrent Neural Networks (RNNs) when trained on a regression task, in this case, a monaural source separation task. The effect of reduced precision nets is explored for two popular recurrent network architectures: Vanilla RNNs and RNNs using Long-Short Term Memory (LSTM) units. The results show that the performance of the networks as measured by blind source separation metrics and speech intelligibility tests on two datasets, show very little decrease even when the weight precision goes down to 4 bits.

***Index Terms*—** Recurrent Neural Network, LSTM RNNs, low precision networks, audio source separation, human-computer interaction

1. INTRODUCTION

Computational auditory scene analysis (CASA) algorithms form one of the state-of-the-art approaches used for addressing monoaural source separation [1, 2] . The main goal of these algorithms is to estimate source-specific masks for separating the individual sources in an audio mixture [3] so that the separated sources include the least amount of undesired interference or distortion. The recent use of Deep Neural Networks (DNNs) to estimate ideal binary masks (IBMs) [4] and ideal ratio masks (IRMs) [5] of the speakers have lead to state-of-the-art results. Although the DNNs used in this task typically have a feedforward architecture, newer DNN models with a recurrent architecture such as Recurrent Neural Networks (RNNs) with simple units (also called vanilla RNNs in this work) and RNNs with gated units such as the Long-Short-Term Memory (LSTM) units [6, 7] have also being investigated for audio tasks which require the extraction of long-and short-time dependencies in the data. RNNs with gated units outperform vanilla RNNs in many tasks including sequence labelling, speech separation, and Automatic Speech Recognition (ASR) [8, 9].

The performance superiority of RNNs comes, however, with a higher computational cost and power; and the presence of a larger number of parameters. For these reasons, RNNs are usually deployed on computing systems that can sustain the computational cost and the memory footprint of these models. One way of reducing the computational cost and memory requirements is to reduce the precision of the DNN parameters. Various studies have demonstrated training methods for reduced precision DNNs that maintain their performance even with reduced precision of the weights in the network [10, 11]. These reduced bit precision training techniques have been applied successfully to DNN architectures such as Deep Belief Networks [12], Convolutional Neural Networks (CNNs) [13], and RNNs [14]. These networks demonstrate very little loss of accuracy compared to the full-precision networks in a classification task.

In this work, we investigate two different aspects of DNNs in the monaural source separation task. The first is the impact of reduced precision weights on the performance of RNNs in this regression task. The second is whether the performance is affected differently for vanilla RNNs and LSTM RNNs. The networks are evaluated on the TIMIT [15] and TSP [16] corpora using SNR computed on the estimated magnitude spectra and blind source separation (BSS) metrics [17] such as source-to-interferences ratio (SIR), source-to-artifacts ratio (SAR) and source-to-distortions ratio (SDR). The quality of the separated sources is also assessed using STOI which measures signal intelligibility [18]. The LSTM RNN architecture for the source separation task is described in Section 2, the experimental setup in Section 3, the results in Section 5 and concluding remarks are presented in Section 6.

2. RELATION TO PRIOR WORK

Vanilla RNNs have previously been used for a single-channel speech separation task where the mixture consisted of two speakers [19, 20] while LSTM RNNs have been used to separate a mixture consisting of noisy speech [9, 21]. Our work differs from previous CASA studies in that we evaluate the impact of the reduced precision parameters of the two

RNN models for the same source separation network. To our knowledge, there has been no work done so far on reduced precision RNNs trained on a regression task and only one previous work describing the training of RNNs with limited bit precision [14].

3. METHODS

3.1. Problem definition

The audio waveforms of the two sources $s_1(t)$ and $s_2(t)$ are first converted to the time-frequency domain using the short-time Fourier transform (STFT). The resulting individual source spectra $\mathbf{S}_1, \mathbf{S}_2 \in \mathbb{R}^{F \times T}$ and the mixture spectrum $\mathbf{S}_m \in \mathbb{R}^{F \times T}$ are used to compute the IRMs for each source separately

$$M_k(t,f) = \frac{|S_k(t,f)|}{|S_1(t,f)| + |S_2(t,f)|} \quad for \quad k \in \{1,2\} \quad (1)$$

where F is the number of frequency bins and N is the number of frames in the magnitude spectra. The two source-specific masks are multiplied elementwise with the magnitude spectra of the mixture to extract the estimated magnitude spectra of the individual sources

$$\left|\hat{\mathbf{S}}_k\right| = \mathbf{M}_k \circ \left|\mathbf{S}_m\right| \quad for \quad k \in \{1,2\} \quad (2)$$

where $\circ$ represents the Hadamard (or elementwise) product. The RNN model is trained in a supervised manner to minimize the difference between the reconstructed magnitude spectrum of each source and its original magnitude spectrum. The separated waveforms are then generated by first applying the phase of the mixture spectrogram to the reconstructed magnitude spectrogram before carrying out the inverse STFT.

3.2. RNN architecture for source separation

The source separation architecture is shown in Figure 1. For both RNN models, the state of each unit in the network depends on both the current input and the previous state of the network. Depending on the type of units used in the network, the update rule for the hidden state is different. For a vanilla RNN, the state of a neuron in layer l receiving input x_t is $h^l(x_t) = f\big(\mathbf{W}^l h^{l-1}(x_t) + \mathbf{U}^l h^l(x_{t-1})\big)$, where $\mathbf{W}^l$ and $\mathbf{U}^l$ are the input and recurrent weight matrices respectively, and $f()$ is a nonlinear function usually a $tanh$. A LSTM RNN has a more complex update rule because each unit has three gates (input gate i_t, output gate o_t, and forget gate f_t) and a candidate memory c_t as shown in Figure 1 (right). The state of the neuron, h_t is a nonlinear function of c_t. The update equations are described in more detail in [6]. The input to our recurrent masking network $\mathbf{S}_m$ goes first to a common RNN (h^1) with B hidden units. The outputs of h^1 drive two separate source-specific RNNs, each with a layer of H neurons. The RNNs of the different branches are h^{2a} for the first source and h^{2b} for the second source. The outputs of these RNNs drive two separate fully-connected layers a^1 and a^2, each consisting of L sigmoidal units. Each fully-connected layer has the same number of units as the frequency dimension of the input magnitude spectrum. The activations are used to estimate the source-specific masks. Note that in this framework, the two masks do not have to sum up to a unity matrix.

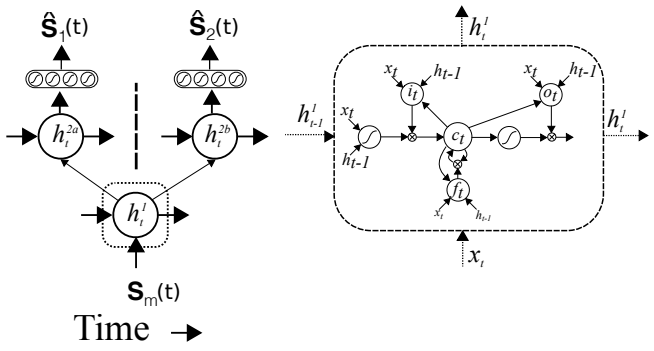

Fig. 1: Network architecture (left) and details of a LSTM unit (right).

3.3. Training

Training to estimate the IRMs of the sources is done using the generalized backpropagation algorithm for recurrent networks. The activations of the output layers (a^1 and a^2) at each time step are used as the masks for the corresponding speaker, that is, $\mathbf{M}_k = \mathbf{a}_N^k \quad for \quad k \in \{1,2\}$, where $\mathbf{a}_N^k \in \mathbb{R}^{F \times N}$ is the activity of a_t^k collected over N frames. The mask is applied to the mixture spectrum $\mathbf{S}_m$, according to Eq. 2, to produce the estimated spectrum $\hat{\mathbf{S}}_k$ of the corresponding speaker. Typically, this estimation is done by minimizing dthe L^2-norm between the estimated and original magnitude spectra.

We use the generalized Kullback-Leibler (KL) divergence in the training cost function. It has been widely used to define the optimization problem of non-negative matrix factorization (NMF), which is a well established method for source separation [22,23]. Importantly for NMF as applied to speech separation, when evaluating the difference between two positive definite high dimensional objects, such as audio spectra, the KL-divergence gives a better estimate than the Euclidean distance [23]. The use of this metric yields the following cost function for our network

$$\mathcal{L}_{KL} = \sum_{t=0}^{N} \sum_{f=0}^{F} \left(\mathbf{S}_k(t,f) \cdot \log \frac{\mathbf{S}_k(t,f)}{\hat{\mathbf{S}}_k(t,f)} - \mathbf{S}_k(t,f) + \hat{\mathbf{S}}_k(t,f) \right) \quad (3)$$

Like the Euclidean distance, the generalized KL-divergence has a lower bound of 0 which is reached if and only if $\hat{\mathbf{S}}_k =$

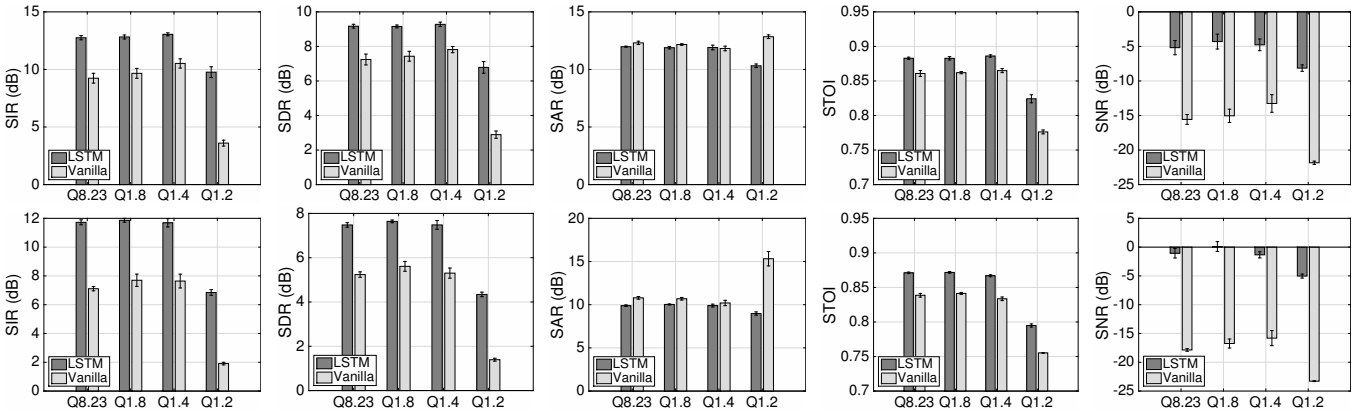

Fig. 2: Results on the TIMIT (top row) and TSP (bottom row) datasets for LSTM RNN and vanilla RNN. The SIR, SDR, SAR, STOI, and SNR scores averaged over all three speaker mixtures M/F, M/M, F/F are presented from left to right . The error bars represent the standard deviation and are obtained with a 10-fold cross validation over the speaker sentences.

$\mathbf{S}_m$, but it cannot be called a "distance" since it is not symmetric.

Each branch of the network estimates the spectra of one of the two sources in the mixture, thus each branch optimizes its own cost function, as described by Eq. 3 for $k \in \{1, 2\}$. Because of the individual branch optimization, the parameter update of the entire network is done in two stages. After the forward pass, a cost is calculated for each branch. In a first step, the parameters of one of the branches are updated, and in a second step, the parameters of the second branch are updated. Therefore, the common RNN layer is updated twice in each training step. Since the updates coming from each branch are additive, the order of branch update does not change the final update.

3.4. Rounding scheme

We describe here the training scheme for the reduced precision networks. The full precision weights of the RNN are expressed with the IEEE 754-2008 single-precision floating-point format that uses 32-bits of which 8 bits are reserved for the exponent, 23 for the fraction and 1 bit for the sign. The format for the reduced precision weights is specified by the fixed-point notation $Qm.f$, where m represents the number of bits in the integer part, including the sign bit, followed by a notational binary point, and f represents the number of bits in the fractional part.

We use the training scheme for rounded weights called *dual-copy rounding* introduced in [12]. With this method, two weight matrices are stored during the training phase of the RNN. One is the high-precision weight matrix (W_H), the other is the low-precision weight matrix (W_L). During training the low precision matrix is used to calculate the forward pass of the network. The parameter updates derived from the backpropagation algorithm, expressed as $\Delta w(W_L)$, are applied to W_H. After the update, both weight matrices are processed following

$$W_L = \begin{cases} -2^m & where \quad W_H \leq -2^m \\ W_H & where \quad -2^m < W_H < 2^m \\ 2^m & where \quad W_H \geq 2^m \end{cases} \tag{4}$$

$$W_L = round(2^f \cdot W_H) \cdot 2^{-f} \tag{5}$$

where 2^m represents the largest possible value of the weight magnitude in the representation $Qm.f$.

The training continues using the updated values of W_L for the next forward pass. We also explored the possibility of rounding the weights only at the end of the training but this decreases significantly the performance of the network.

4. EXPERIMENTAL SETUP

4.1. Data preprocessing

The TIMIT and TSP datasets are used in the validation of the model. For each dataset, three combinations of speaker pairs: Male/Female (M/F), Male/Male (M/M) and Female/Female (F/F) speakers are considered. In all cases, 80% of the sentences are used for training, 10% for validation, and 10% for testing. For each speaker, the training sentences are concatenated to create one training vector. The training vector of one speaker is then added to the training vector of the second speaker using 10 random shift values. This operation increases the number of mixture training sentences by 10 times leading to 45 minutes of training data. The audio waveforms of the individual speakers and the mixture are first transformed into spectrograms using a 512 point STFT and an overlap of 75% between successive Hanning windows. Each training and test mixture spectrogram sample is of size $F \times N$ where $F = 257$ and $N = 50$. Since 512 samples of the raw audio signal are used for each frame, every spectra sample of

50 frames corresponds to 0.424s of the audio signal which is sampled at 16kHz.

4.2. Network specifications

The networks in our simulations have the following parameters. The first layer h^1 has B hidden units. The individual recurrent layers h^{2a} and h^{2b} have H hidden units. The fully-connected layers a^1 and a^2 have $L = 257$ sigmoidal units. Because a LSTM RNN has 4 times the number of parameters of a vanilla RNN, we kept the total number of parameters in both networks fixed to $P = 500000$ so that a fair comparison between the two networks can be made. This leads to a LSTM RNN with $B = H = 123$ units and a vanilla RNN with $B = H = 248$ units. The networks are trained for 200 epochs, where every epoch is composed of 10 mini-batches of 150 training examples each. The training is done using Tensorflow and optimization is done using the gradient descent algorithm, Adam [24]. Validation is done using the spectrogram SNR, BSS-eval metrics, and STOI tests.

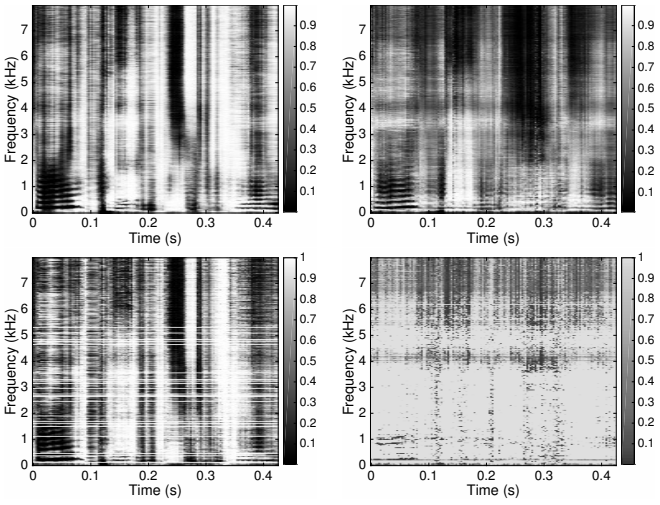

Fig. 3: Masks for a speech sample from speaker MC of the TSP dataset generated by a LSTM RNN (top) and a vanilla RNN (bottom). Different bit-precision values were used, $Q8.23$ (left column) and $Q1.2$ (right column).

5. RESULTS

Figure 2 summarizes the results from the different bit-precision networks averaged over the different speaker mixtures M/F, M/M and F/F. Because the separation of speakers of different genders (M/F) yields better results than when the speakers are of the same gender, we averaged the scores over all pairings. The results show that both networks show little drop in performance even with precision down to $Q1.4$, therefore demonstrating that regression networks can also be trained with low precision weights just as with deep networks trained on classification tasks. It is important to use a scheme that rounds weights during training. Rounding the weights at $Q1.8$ after training yields a drop of 25% in SIR and SDR and a drop of 3% in STOI in both datasets and for both LSTM and Vanilla RNN. Even with full precision parameters ($Q8.23$), a LSTM RNN produces a better estimation of the separated magnitude spectra than the vanilla RNN. This is because the estimated masks from the LSTM RNN do not show artifacts that can be clearly seen in the masks estimated by the vanilla RNN (see the strips in certain frequency bands for an example in Figure 3 (left)). On both data sets, the LSTM RNN has on average, a significant SIR gain of 2.5 dB and SDR gain of 2 dB over the vanilla RNN. No significant difference can be found between the SAR scores for both networks. With respect to the full-precision network, the $Q1.2$ LSTM RNN has a performance drop of 7% for the STOI scores (~ 0.8) on the TSP database. Nevertheless since the STOI score for the original mixture is 0.75, the result shows that this model still enhances the intelligibility of single sources in a mixture even with low precision weight and bias parameters. Conversely, the $Q1.2$ vanilla RNN has a performance drop of almost 10% on the TSP database compared to the full-precision network, reaching a score which indicates that there is no increase in intelligibility with respect to the original mixture. The SIR and SDR drops of about 30% for the LSTM RNN and about 60% for the vanilla RNN show that even the $Q1.2$ LSTM RNN gives comparable SIR and SDR scores with respect to a vanilla RNN trained with full-precision weights on both datasets. As can be seen in Figure 3 (right), a $Q1.2$ LSTM RNN still produces a mask which retains the spectral pattern of the speech while the $Q1.2$ vanilla RNN mask has smeared frequency bands.

6. CONCLUSION

This paper presents a study of the impact of reduced precision on deep regression RNNs. The results show that the full precision vanilla RNN and LSTM RNN have similar performance numbers in the monaural source separation task and that their performances degrade very slowly even when the bit precision of the weights and biases are reduced to 4 bits. The LSTM RNN however shows a smaller drop in performance compared to the vanilla RNN. The results will help guide the implementation of RNN models in embedded systems which are faced by limited computing resources and a limited power budget.

7. ACKNOWLEDGEMENTS

We acknowledge D. Neil, S. Braun, I. Kiselev, and S. Bhargava for useful discussions on RNN models, source separation and audio processing. This work was partially supported by EU H2020 COCOHA # 644732.

8. REFERENCES

[1] D. Wang and G. J. Brown, *Computational auditory scene analysis: Principles, algorithms, and applications*, Wiley-IEEE Press, 2006.

[2] A. S. Bregman, *Auditory Scene Analysis: The Perceptual Organization of Sound*, MIT Press, 1990.

[3] DeLiang Wang, "On ideal binary mask as the computational goal of auditory scene analysis," in *Speech Separation by Humans and Machines*, pp. 181–197. Springer US, 2005.

[4] X. Zhao, Y. Wang, and D. Wang, "Robust speaker identification in noisy and reverberant conditions," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 836–845, 2014.

[5] A. Narayanan and DeLiang Wang, "Investigation of speech separation as a front-end for noise robust speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 826–835, 2014.

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] K. Cho, B. Van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Oct. 2014, pp. 1724–1734.

[8] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 2067–2075.

[9] F. Weninger, F. Eyben, and B. Schuller, "Single-channel speech separation with memory-enhanced recurrent neural networks," in *Acoustics, Speech and Signal Processing (ICASSP)*. 2014, pp. 3709–3713, IEEE.

[10] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Advances in Neural Information Processing Systems 28*, pp. 3105–3113. Curran Associates, Inc., 2015.

[11] Z. Lin, M. Courbariaux, R. Memisevic, and Y. Bengio, "Neural networks with few multiplications," *ICLR 2016*, 2015.

[12] E. Stromatias, D. Neil, M. Pfeiffer, F. Galluppi, S. B. Furber, and S.-C. Liu, "Robustness of spiking Deep Belief Networks to noise and reduced bit precision of neuro-inspired hardware platforms," *Frontiers in Neuroscience*, vol. 9, pp. 222, jul 2015.

[13] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," *CoRR*, vol. abs/1603.05279, 2016.

[14] J. Ott, Z. Lin, Y. Zhang, S.-C. Liu, and Y. Bengio, "Recurrent Neural Networks With Limited Numerical Precision," *ArXiv e-prints*, Aug. 2016.

[15] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1," vol. 93, 1993.

[16] P. Kabal, "TSP Speech Database," 2002.

[17] E. Vincent, R. Gribonval, and C. Fvotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[18] C.H. Taal, R.C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time #x2013;frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[19] P. Huang, M. Kim, M. H., and P. Smaragdis, "Deep learning for monaural speech separation," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1562–1566.

[20] P. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 12, pp. 2136–2147, 2015.

[21] F. Weninger, J. R Hershey, and J. L. Roux, "Discriminatively Trained Recurrent Neural Networks for Single-Channel Speech Separation," *GlobalSIP*, pp. 577–581, 2014.

[22] D. Lee and H. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, , no. 1, pp. 556–562, 2001.

[23] M. N. Schmidt, "Speech separation using non-negative features and sparse non-negative matrix factorization," in *Interspeech*, 2006.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.