Transformer Model Compression for End-to-End Speech Recognition on Mobile Devices

Leila Ben Letaifa
LaBRI, CNRS UMR 5800
Univ. de Bordeaux, Bordeaux INP
Talence, France
leila.ben-letaifa@labri.fr

Jean-Luc Rouas
LaBRI, CNRS UMR 5800
Univ. de Bordeaux, Bordeaux INP
Talence, France
jean-luc.rouas@labri.fr

***Abstract*—Transformer-based models have achieved state-of-the-art performance in various areas of machine learning, including automatic speech recognition. However, their cost in terms of computational power, memory or energy consumption can be exorbitant, hence the interest in compression techniques. Transformer models are mostly composed of attention and feedforward components. In this paper, we propose to reduce the size of a transformer model in an end-to-end speech recognition system by decreasing the number and precision of linear layer parameters. Specifically, we investigate the impact of weight pruning on system performance. We then consider model quantization. To further reduce the model size, we address the combination of pruning and quantization methods. Experiments carried out on several speech datasets from different languages show that the memory footprint can be reduced by up to 84% with an insignificant loss of accuracy.**

***Index Terms*—End-to-end speech recognition, transformer model, compression techniques, quantization, pruning.**

I. Introduction

Deep learning has revolutionized several fields of information systems, including natural language processing, image classification and speech recognition. Traditional Automatic Speech Recognition (ASR) systems are based either on hidden Markov models [1] or on hybrid models derived from the combination of neural networks and Markov models [2]. They involve the tuning of several modules separately: acoustic model, lexicon and language model. Through deep learning, the functionality of all these components can be integrated into a single neural network, leading to end-to-end ASR systems. Several kinds of end-to-end (E2E) models have been reported in the literature: Connectionist Temporal Classification based (CTC) [3], attention-based encoder-decoder [4], joint CTC/attention based [5] and RNN transducer [6].

Optimization of machine learning models is a recurrent problem [7] [8]. Indeed, the computing resources available on the market are increasing day by day leading to the tuning of larger and often more accurate models on powerful computers. However, the devices have more limited hardware resources in terms of memory, computation and battery [9]. They usually need smaller models to ensure low latency.

Neural network compression methods include quantization [10], pruning [11], knowledge distillation [12], matrix decomposition [13] and parameter sharing [14]. Although most of these methods were originally proposed for convolutional neural networks, some of them are directly applicable to transformer model. Compared to basic models such as convolutional or recurrent neural networks, a transformer model [15] has a relatively complex architecture composed of several parts such as embedding layers, self-attention layers and feedforward layers. Thus, the effect of compression methods can vary when applied to different parts of it [14].

Research on transformer model compression in E2E speech recognition has mainly focused on quantization [16] and parameter sharing [17]. In this work, we extend it to weight magnitude pruning. We will then discuss quantization. Finally, the combination of both pruning and quantization approaches is considered to further reduce the memory footprint.

The reminder of this paper is as follows: Section 2 reviews compression techniques. Section 3 introduces briefly the ASR transformer model. Section 4 presents our model compression experiments. Section 5 discusses the experimental results and Section 6 draws conclusions and describe future directions.

II. Compression techniques

Model compression methods reduce the inference costs of trained models. In particular, we consider two compression techniques: quantization and pruning.

A. Quantization

By default, most systems uses float 32 types to represent variables and weights. Quantization replaces floating points with integers leading to less memory consumption and faster calculations on certain hardware [10].
Quantization maps a floating point value x in $[a, b]$ to a $k-bit$ integer x_q in $[-2^{k-1}, 2^{k-1} - 1]$.

$$x_q = round(\frac{x - a}{\delta})$$

where $\delta = \frac{b-a}{2^k-1}$. In the case that x in not in the range of $[a, b]$, the clamp operator is applied :

$$clamp(x; a, b) = min(max(x, a), b)$$

The de-quantization function is defined as :

$$D(x_q) = x_q * \delta + a$$

Different quantization approaches have been proposed and can be classified into two categories: post-training quantification and quantification aware training [16].

B. Pruning

Deep learning models are often over-parameterized [11] [18]. They have many insignificant weights that contribute very little to the inference of the model. These weights can be set to zero without significantly affecting performance [11]. The importance of the weights can be determined by their magnitude, their gradients or a custom measurement [14]. There are two types of pruning:

- unstructured pruning [18], which removes individual weights. Substituting the value of the weight with zero in a weight matrix is equivalent to pruning a connection. This pruning is also called sparse pruning because it results in sparse matrices.
- structured pruning [19] focuses on pruning blocks of weights, for example, by deleting entire channels at a time. In practice, structured pruning sets an entire row or column of a weight matrix to zero, which is the same as deleting a neuron.

Pruning can be incorporated into the training process as an additional step between training epochs (iterative pruning), applied all at once after the model training is complete (one-shot pruning), or applied between fine tuning steps.
Whether the pruning is applied globally to all model parameters or is calculated independently for each layer, it is called global pruning or local pruning. Global pruning groups all the parameters of the layers and selects a global fraction of them to prune. Local pruning removes a fixed percentage of parameters from each layer.

III. Transformer model

Transformer [20] is a sequence-to-sequence model that maps an input sequence of acoustic features $(x_1, x_2, ...x_T)$ of length T to an output sequence of characters $(y_1, y_2, .., y_L)$ of length U. Its architecture can be divided into two parts namely the encoder and the decoder. The encoder converts the input sequence into an intermediate sequence of encoded features $(h_1, h_2, ..., h_N)$ of length N. The decoder predicts a new character y_l based on the encoded features $(h_1, h_2, ..., h_N)$ and the previous decoded characters $(y_1, y_2, ..., y_{l-1})$. Both the encoder and the decoder are composed of a stack of attention and feedforward network blocks.

Our ASR transformer follows the same architecture as [21]. A simplified representation of its main components is shown on Figure 1. Here, the input acoustic features are subsampled using two convolution layers before being fed into the encoder.

IV. Experiments and results

We develop transformer-based ASR systems, then prune and quantify the models and measure the performance. We consider the trade-off between error and compression rates. In quantization, the compression rate is the gain in model storage space, whereas in pruning, it is rather the sparsity. To compare

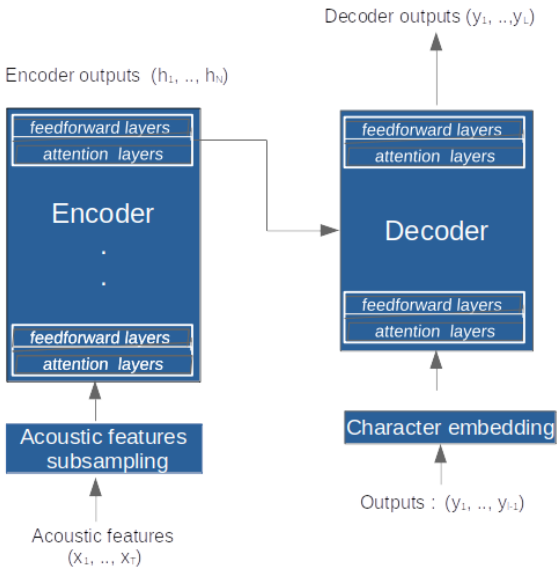

Fig. 1. ASR Transformer main components

the results of the different methods, we propose a compression rate based on doubly compressed models.

A. Data description

Four corpora from different languages, which are English, French, Italian and Vietnamese, are used for all experiments. They are respectively named Libri-trans [22], Ester [23], Voxforge [24] and Vivos [25]. Libri-trans is a part ($\sim$ 236 hours) of the LibriSpeech dataset, comprising annotated English audio books. The Ester corpus contains transcripts of about 250 hours of broadcast news produced in the scope of the French national ESTER project. The Voxforge project has also created several databases including 20 hours of Italian audiobooks forming VoxforgeIt. 16 hours of reading texts in Vietnamese make up the Vivos database. The training part of each dataset is employed for the parameters estimation and the rest for the evaluation. This is around 3.5 hours for Libri-trans, 6.5 hours for Ester and 1 hour for each of Voxforge and Vivos.

B. Baseline systems

Baseline end-to-end ASR systems are developed with the Espnet toolkit [5]. In order to augment the amount of data, their speed is perturbed. By using three different speeds, the train dataset amount tripled. Then, 80 f-bank coefficients are extracted and normalized with respect to the mean and variance. Transcripts are represented by subword units, namely characters for the Ester, Voxforge, and Vivos systems and byte-pair coding subwords for the Libritrans system. Finally, several transformer architectures are evaluated. Table I shows the architecture of the best transformer models, their memory size (in megabytes), their number of parameters (in millions) and the error rates of the ASR systems. We consider the word errors (WER) of the Libri-trans and Ester systems and the character errors (CER) of the Voxforge and Vivos systems.

TABLE I

ASR MODELS SPECIFICATIONS: - ARCHITECTURE : NUMBER OF ENCODER AND DECODER BLOCKS (ENC/DEC), DIMENSION OF HIDDEN LAYERS (FF DIM) AND ATTENTION LAYERS (ATT DIM) AND NUMBER OF ATTENTION HEADS (HEADS) - MEMORY SIZE (MB) - NUMBER OF PARAMETERS (MILLIONS) - ERROR RATE (% WER/CER)

	LIBRITRANS	ESTER	VOXFORGE	VIVOS
ARCHITECTURE				
ENC/DEC	12/6	18/6	12/6	8/2
FF DIM	1024	2048	2048	1024
ATT DIM	256	512	256	256
HEADS	4	4	4	4
MEMORY SIZE	107	343	134	87
PARAMETERS	27.92	89.64	35.07	15.65
ERROR RATE	6.6	14.1	9.1	14.7

C. Parameter distribution

Given the architecture of our transformer models, we roughly distinguish two kinds of layers: convolution layers and linear layers. The convolution layers are present in the subsampling block. All other layers, i.e. input, attention, feedforward and output layers are linear layers.

TABLE II

DISTRIBUTION OF THE PARAMETERS BETWEEN THE CONVOLUTION LAYERS (CONV) AND THE LINEAR LAYERS OF THE ENCODER (ENCODER) AND THE LINEAR LAYERS OF THE DECODER (DECODER).

	CONV (%)	ENCODER (%)	DECODER (%)
LIBRI-TRANS	2.12	61.15	34.80
ESTER	2.63	69.10	28.14
VOXFORGE	1.68	71.16	27.00
VIVOS	3.78	75.52	20.30

Table II reports the percentage of parameters in the convolutional layers, in the encoder and in the decoder. Since the encoder and decoder contain the majority of the parameters, we decided not to decrease the parameters of the convolutional layers. In the following, the compression techniques will be applied to the linear layers.

D. Pruning

One-shot pruning is applied to the trained models. The pruning type is set to global or local. Then we vary the pruning rate and report the recognition error rate. For the best trade-off between error and compression rates, we tolerate a relative increase in error rate of 10% . According to Fig. 2, the best trade-offs are achieved with pruning rates of 33% for Libri-trans, 37% for Ester, 55% for Voxforge and 62% for Vivos. The pruning rate of a model is closely related to its size when compressed by a classical method (such as gzip). Indeed, classical methods are effective in reducing models containing null strings. In addition, they are useful for storage on mobile devices. Here, gzip compression is applied for comparison purposes; the initial and pruned models are gzipped and their ratio is calculated. The pruned models are those with pruning ratios satisfying the tradeoffs. Table III shows that pruning can reduce the size of compressed models by half.

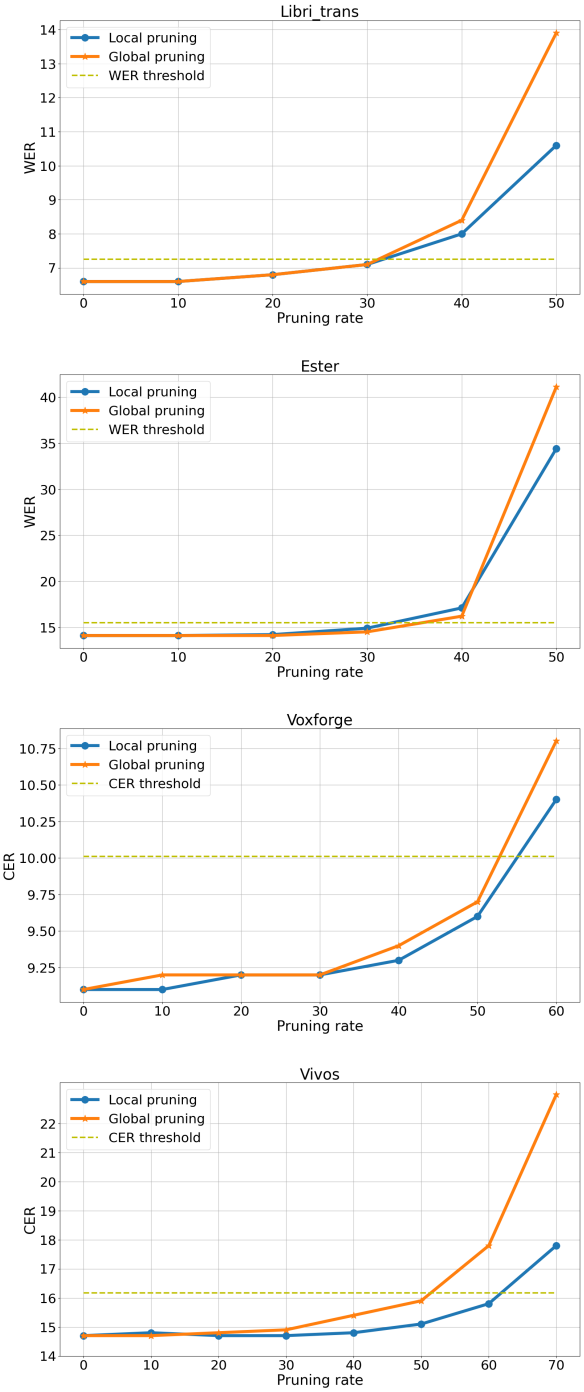

Fig. 2. Error rate as a function of local and global pruning rate for the four ASR systems: Libri-trans, Ester, Voxforge and Vivos

TABLE III

SIZE OF INITIAL GZIP-COMPRESSED AND PRUNED GZIP-COMPRESSED MODELS (IN MEGABYTES) AND THEIR RATIO (PRUNE.GZ/INIT.GZ)

	LIBRITRANS	ESTER	VOXFORGE	VIVOS
INIT.GZ (MB)	99	318	124	80
PRUNE.GZ (MB)	72	228	72	41
RATIO (%)	72.72	71.69	58.06	51.25

E. Quantization

Initially all model parameters are stored and processed in real 32-bit format. An 8-bit integer quantization is performed on the encoder and decoder weight matrices. Activations are quantized on the fly during inference and stored as 32-bit real values.

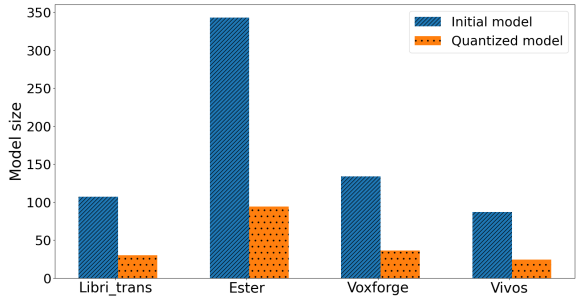

Fig. 3. Size of the initial and quantized models.

For all systems, the performance decrease is insignificant (less than 0.1%). Regarding the model size, Fig. 3 shows that quantized models are more than 70% smaller.
The initial and quantized models are then compressed by gzip.

TABLE IV
SIZE OF GZIP-COMPRESSED QUANTIZED MODELS (IN MEGABYTES) AND THEIR RATIO TO THE INITIAL GZIP-COMPRESSED MODELS.

	LIBRITRANS	ESTER	VOXFORGE	VIVOS
QUANT.GZ (MB)	23	72	29	21
RATIO (%)	23.23	22.64	23.38	26.25

According to Table IV, the quantized models are about four times smaller than the initial models.

F. Pruning then Quantization

Pruning and quantization are now applied successively for a better accuracy/size trade-off. This is performed in three steps.

1) set the pruning rate
2) prune the model
3) quantize the model

The initial and final models are then gzipped and their size ratios are computed to assess the joint contribution of pruning and quantization. We plot the model size ratios and error rates in Fig. 4 and derive the best trade-off. The error rate threshold is set to 1.1 * error-rate. This threshold is associated with a certain pruning rate which in turn induces the compression rate of the trade-off (it is highlighted by a large orange dot).

The pruning and compression rates of the best trade-offs are reported in Table V. We note that the pruning rate recorded is generally slightly lower than that for pruning alone (see Fig. 2). However, the compression ratio of all models is better than that of pruning and quantization alone (Table III and Table IV). A final compression of about 84% for Vivos, 82% for Voxforge, 80% for Libri-trans and 79% for Ester is achieved.

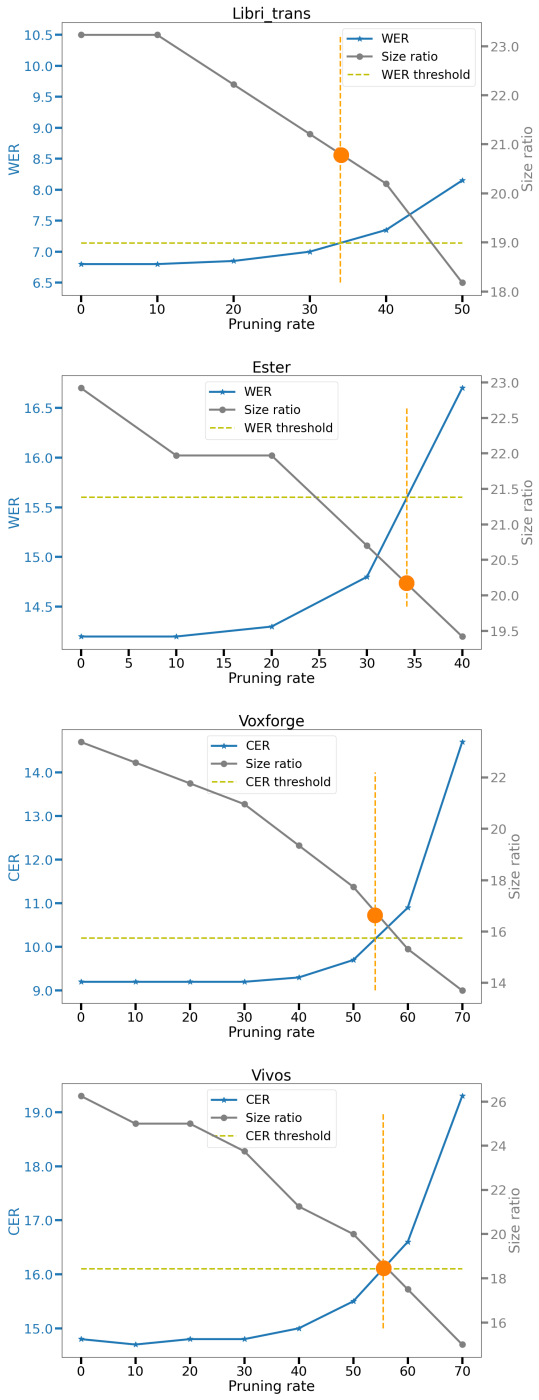

Fig. 4. Pruning and quantization of Libri-trans, Ester, Voxforge and Vivos models: Error and compression rates as a function of pruning rate

V. DISCUSSION

Structured pruning leads to 62% sparser models. Quantization from float32 to int8 divides the model size by four. Reducing the accuracy of sparse matrices, i.e., combining

TABLE V
INITIAL AND FINAL ERROR RATES, PRUNING RATE, SIZE OF INITIAL GZIP-COMPRESSED AND PRUNED+QUANTIZED GZIP-COMPRESSED MODELS (IN MEGABYTES) AND THEIR RATIO (FINAL.GZ/INIT.GZ)

	LIBRITRANS	ESTER	VOXFORGE	VIVOS
INIT. ERROR (%)	6.6	14.1	9.1	14.7
FINAL ERROR (%)	7.2	15.6	10.1	16.1
INIT.GZ (MB)	99.0	318.0	124.0	80.0
PRUNING RATE (%)	33.4	34.5	53.5	56.0
FINAL.GZ (MB)	20.2	68.0	20.5	14.7
RATIO (%)	20.5	21.4	16.6	18.4

pruning and quantization, takes advantage of both methods and achieves a compression of about 84%.

The Voxforge and Vivos systems have the highest pruning rates. A closer look at these systems reveals that their models are large and their databases are small. In fact, Voxforge and Vivos models have sizes comparable to that of Libri-trans (see Table I). At the same time, they are trained on more than 10 times smaller amount of data; they are obviously more over-parameterized. This is not reflected by quantization, which reduces the accuracy of all weights regardless of their values (very small or not).

Although the Libri-trans and Ester datasets are nearly the same size, the Ester models are larger. They are also less well compressed. This can be due to the difference in language. French speech can have higher variability, which implies a larger and denser model.

VI. CONCLUSION AND FUTURE WORK

In this paper, we explore weight pruning of transformer models in end-to-end speech recognition. This research is carried out on four different language systems: English, French, Italian and Vietnamese. We realize that the more over-parameterized the models are the more pruned they are, leading to about 62% more sparse matrices. Next, we evaluate the impact of degrading the precision of the models by quantization and we achieve a fourfold reduced model memory footprint for an insignificant increase in error rate. To further downsize the model, we lower the precision of the pruned models. This results in an overall model compression of about 84%. Future work includes fine-tuning the models and exploring the combination with other compression techniques.

ACKNOWLEDGMENT

The research presented in this paper is conducted as part of the project FVLLMONTI that have received funding from the European Union's Horizon 2020 Research and Innovation action under grant agreement No 101016776.

REFERENCES

[1] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[2] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on audio, speech and language processing*, 2012.

[3] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," 2006.

[4] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," in *Deep Learning and Representation Learning Workshop, NIPS*, 2014.

[5] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "Espnet: End-to-end speech processing toolkit." Proceedings of Interspeech, 2018, pp. 2207–2211.

[6] F. Boyer, Y. Shinohara, T. Ishii, H. Inaguma, and S. Watanabe, "A study of transducer based end-to-end asr with espnet: Architecture, auxiliary loss and decoding strategies," in *Proceedings ASRU*, 2021.

[7] L. Zouari and G. Chollet, "Efficient codebooks for fast and accurate low resource asr systems," *Speech Communication*, vol. 51, pp. 732–743, 2009.

[8] L. Zouari and Chollet, "Multi-level gaussian selection for accurate low-resource asr systems," in *International Conference on Human Language Technologies*, 2007.

[9] L. Beltaifa-Zouari, "Embedded real time speech recognition system for smart home environment," *IJSER*, 2017.

[10] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *The Journal of Machine Learning Research*, vol. 18, p. 6869–6898, 2017.

[11] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems*, 1989, pp. 598–60.

[12] H.-G. Kim, H. Na, H. Lee, J. Lee, T. G. Kang, M.-J. Lee, and Y. S. Choi, "Knowledge distillation using output errors for self-attention end-to-end models," in *Proceedings ICCASP*, 2019.

[13] M. B. Noach and Y. Goldberg, "Compressing pre-trained language models by matrix decomposition," in *International Joint Conference on Natural Language Processing*, 2020.

[14] P. Ganesh, Y. Chen, X. Lou, M. A. Khan, Y. Yang, H. Sajjad, P. Nakov, D. Chen, and M. Winslett, "Compressing large-scale transformer-based models: A case study on bert," *Transactions of the Association for Computational Linguistics*, vol. 9, 2021.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[16] A. Bie, B. Venkitesh, J. Monteiro, M. A. Haidar, and M. Rezagholizadeh, "A simplified fully quantized transformer for end-to-end speech recognition," 2020. [Online]. Available: https://arxiv.org/abs/1911.03604

[17] S. Li, D. Raj, X. Lu, P. Shen, T. Kawahara, and H. Kawai, "Improving transformer-based speech recognition systems with compressed structure and speech attributes augmentation." Graz, Austria: Proceedings of Interspeech, 2019, pp. 4400–4404.

[18] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Advances in Neural Information Processing Systems*, 2015.

[19] S. Anwar, K. Hwang, and W. Sung, "Structured pruning of deep convolutional neural networks," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 1, p. 1–18, 2017.

[20] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs rnn in speech applications," in *ASRU*, 2019.

[21] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition." Proceedings the International Conference on Acoustics, Speech, and Signal Processing, 2018.

[22] A. C. Kocabiyikoglu, L. Besacier, and O. Kraif, "Augmenting librispeech with french translations: A multimodal corpus for direct speech translation evaluation," in *LREC*, 2018.

[23] S. Galliano, G. Gravier, and L. Chaubard, "The ester 2 evaluation campaign for the rich transcription of french radio broadcasts," in *Interspeech*, 2009, p. 2583–2586.

[24] "Voxforge (italian). http://www.voxforge.org," 2019.

[25] H.-T. Luong and H.-Q. Vu, "A non-expert kaldi recipe for vietnamese speech recognition system," in *WLSI/OIAF4HLT*, 2016.