# Fine-grained analysis of the transformer model for efficient pruning

Leila Ben Letaifa, Jean-Luc Rouas

# Fine-grained analysis of the transformer model for efficient pruning

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*—In automatic speech recognition, deep learning models such as transformers are increasingly used for their high performance. However, they suffer from their large size, which makes it very difficult to use them in real contexts. Hence the idea of pruning them. Conventional pruning methods are not optimal and sometimes not efficient since they operate blindly without taking into account the nature of the layers or their number of parameters or their distribution. In this work, we propose to perform a fine-grained analysis of the transformer model layers in order to determine the most efficient pruning approach. We show that it is more appropriate to prune some layers than others and underline the importance of knowing the behavior of the layers to choose the pruning approach.**

***Index Terms*—Speech recognition, transformer model, pruning techniques, weight magnitude, model analysis**

## I. Introduction

In recent times, information system fields such as language processing [1], image analysis [2], speech recognition [3] and emotion detection [4] have made great progress with deep learning models. In automatic speech recognition (ASR), we refer to end-to-end (E2E) systems. For machine learning models, the number of parameters and performance are often correlated [5]. In particular, transformers, which are state-of-the-art models, are very resource-intensive in terms of computing power, memory, energy consumption,.. [6]. Furthermore, it was shown that large, over-parameterized models are more accurate than small and dense models [7]. For these reasons, model compression techniques are required.

Neural network compression methods fall into several categories, which are quantization [8], pruning [9], knowledge distillation [10], matrix decomposition [11] and parameter sharing [12]. Compared to basic models such as recurrent neural networks (RNN) or multilayer perceptron (MLP), a transformer model has a relatively complex architecture composed of several parts such as embedding layers, multi-head attention layers and feedforward layers. Thus, the effect of compression methods can vary when applied to different parts of a transformer model [12]. Research on the compression of transformer models in E2E speech recognition has tackled the problem of quantization [13], parameter sharing [14] and recently pruning [15] using the conventional techniques. This research investigates the weight pruning for ASR as mean of transformer model compression.

Global pruning and local pruning are the two conventional orthogonal pruning schemes [16] [17]. While local pruning prunes every layer of a model with the same rate, global pruning considers the model as a whole and prunes the lowest parameters [18]. One of the drawbacks of these methods is that they work blindly, regardless of the type of layers, their number of parameters or even their behaviour. In this paper, we proceed to a fine-grained analysis of transformer model. We show that some layers such as the convolution layers have a very small number of parameters and pruning them is useless and decreases drastically the performance. We also highlight the behavior of certain layers such as the feedforward or multi-head attention layers that may be relevant to the choice of pruning technique.

The remainder of this paper is as follows: Section 2 reviews the pruning techniques. Section 3 introduces briefly the ASR transformer model and Section 4 presents our trained models. Section 5 describes the layers' parameters and behavior. Section 6 presents the pruning experiments and results and Section 7 draws conclusions and proposes future directions.

## II. Pruning schemes

Deep learning models have many insignificant weights that contribute very little to the inference of the model [9] [2]. These weights can be set to zero without significantly affecting performance [9]: This is called model pruning. The significance of these weights can be determined by their magnitude, their gradients or a custom measurement [12]. 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) [15], or applied between fine tuning steps [16]. There are two conventional pruning schemes [17]: Global pruning and local pruning. Global pruning, also called class-blind pruning [19], gathers all layer parameters and selects a global fraction of them to prune. Local pruning (i.e. class-uniform pruning) removes a fixed percentage of parameters from each layer. These methods are not based on a prior analysis of the layers: they do not take into account the nature of the layers, their position in the network, their number of parameters or even the relationships between them.

## III. ASR Transformer models

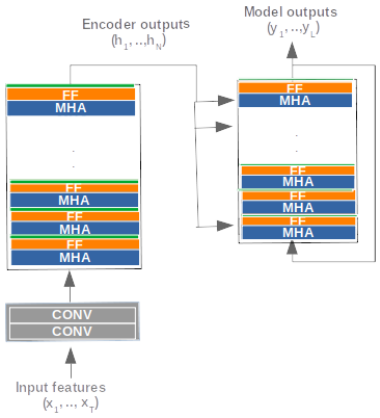
The transformer model [1] is a sequence-to-sequence model that maps an input sequence to an output sequence.

### A. Model description

The ASR transformer model takes as input a sequence of acoustic features $(x_1, x_2, ..., x_T)$ and generates a set of characters $(y_1, y_2, ..., y_L)$, one character at a time. At each step, the model is auto-regressive, taking previously generated characters as an additional input when generating the next character [1]. 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 $(h_1, h_2, ..., h_N)$. The decoder predicts a new character $y_l$ based on the encoded features $(h_1, h_2, ..., h_N)$ and the previous decoded characters $(y_1, y_2, ..., y_{l-1})$.

Our ASR transformer follows the same architecture as [20]. The input acoustic features are subsampled using two convolution layers (CONV) before being fed into the encoder. Both the encoder and the decoder are composed of multi-head attention (MHA) and feedforward (FF) layers, each followed by a residual connection and normalization. A simplified representation of the transformer model is shown on Fig. 1.

Fig. 1: ASR Transformer main components that are the convolution layers (CONV), the multi-head attention (MHA) layers and the feedforward (FF) layers.




### B. Model architecture

The self-attention operation allows frames to gather context from all timesteps and build an informative sequence of high level [20]. Specifically the the inputs of each layer are projected into queries $Q$, keys $K$ and values $V$ with $Q \in \mathbb{R}^{t_q * d_q}$, $K \in \mathbb{R}^{t_k * d_k}$ and $V \in \mathbb{R}^{t_v * d_v}$. $t*$ are the elements numbers in different inputs and $d*$ are the corresponding element dimensions. Scaled Dot-Product Attention [1] is then computed as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

The multi-headed attention is obtained by performing this calculation $h$ times. $h$ is the number of heads.

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W_0 \quad (2)$$

where

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

The projection matrices are $W_i^Q \in \mathbb{R}^{d_{model} * d_q}$, $W_i^K \in \mathbb{R}^{d_{model} * d_k}$, $W_i^V \in \mathbb{R}^{d_{model} * d_v}$ and $W^0 \in \mathbb{R}^{h * d_v * d_{model}}$. In this work, $d_k = d_q = d_v = d_{model}/h$

The outputs of multi-head attention go through 2-layer position-wise feedforward network (FFN) with hidden size $d_{ff}$.

$$FFN(x) = W_2 ReLU(W_1 x + b_1) + b_2 \quad (4)$$

$b_1 \in \mathbb{R}^{d_{ff}}$ and $b_2 \in \mathbb{R}^{d_{model}}$ are the biases. The weight matrices are $W_1 \in \mathbb{R}^{d_{model} * d_{ff}}$ and $W_2 \in \mathbb{R}^{d_{ff} * d_{model}}$.

## IV. Baseline models

We have developed transformer models for three languages which are English, French and Italian using Libri-trans [21], Ester [22], and Voxforge [23] databases respectively.

### A. Data description

The Libri-trans, Ester and Voxforge corpora are produced within the framework of the Librivox project, the French national ESTER project and the Voxforge project. The Libri-trans and Voxforge recordings are extracted from audiobooks, and the Ester recordings are radio broadcasts news. Each dataset is divided into three parts: train, development (dev) and test as described in Table I. The train data is used for model training. The dev and the test parts are dedicated to evaluation.

TABLE I: Duration (in hours) of the train, dev. and test parts of the three datasets.

| | Libritrans | Ester | Voxforge |
|---|---|---|---|
| Train | 230 | 231 | 18 |
| Dev | 2 | 5.45 | 1 |
| Test | 3.5 | 6.5 | 1 |

### B. Trained models

Baseline ASR transformer models are developed and evaluated with the Espnet toolkit [3]. This toolkit involves Kaldi [24] tools for data processing and parameter extraction and Pytorch (pytorch.org) modules for model estimation. First, by using three different speeds (0.9, 1.0, and 1.1), the train dataset amount tripled. Then, 80 filter bank coefficients are extracted and normalized with respect to the mean and variance. Transcripts are represented by sub-word units, namely characters for the Ester and Voxforge systems and byte-pair coding subwords for the Libritrans system. Finally, several transformer architectures are evaluated. Table II shows the architecture of the best transformer models, 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 system.

TABLE II: 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) - Number of parameters (Millions) - Error rate (% WER/CER)

| | LIBRITRANS | ESTER | VOXFORGE |
|---|---|---|---|
| ARCHITECTURE | | | |
| ENC/DEC | 12/6 | 18/6 | 12/6 |
| FF DIM | 1024 | 2048 | 2048 |
| ATT DIM | 256 | 512 | 256 |
| HEADS | 4 | 4 | 4 |
| PARAMETERS | 27.92 | 89.64 | 35.07 |
| ERROR RATE | 6.6 | 14.1 | 9.1 |

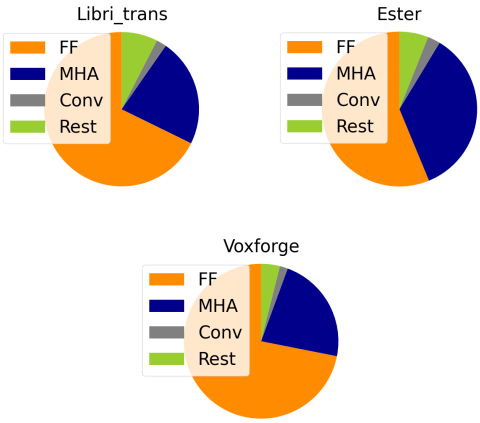
## V. MODEL WEIGHTS ANALYSIS

The layers of the transformer model are organized into four groups: CONV layers, MHA layers, FF layers, and the remaining layers.

### A. Number of parameters

The proportion in number of weights of each group is calculated and then plotted in Fig. 2.

Fig. 2: The proportion of weights of convolution layers (Conv), multi-head attention layers (MHA), feedforward layers (FF) and remaining layers (Rest) for the Libri-trans, Ester, and Voxforge models.

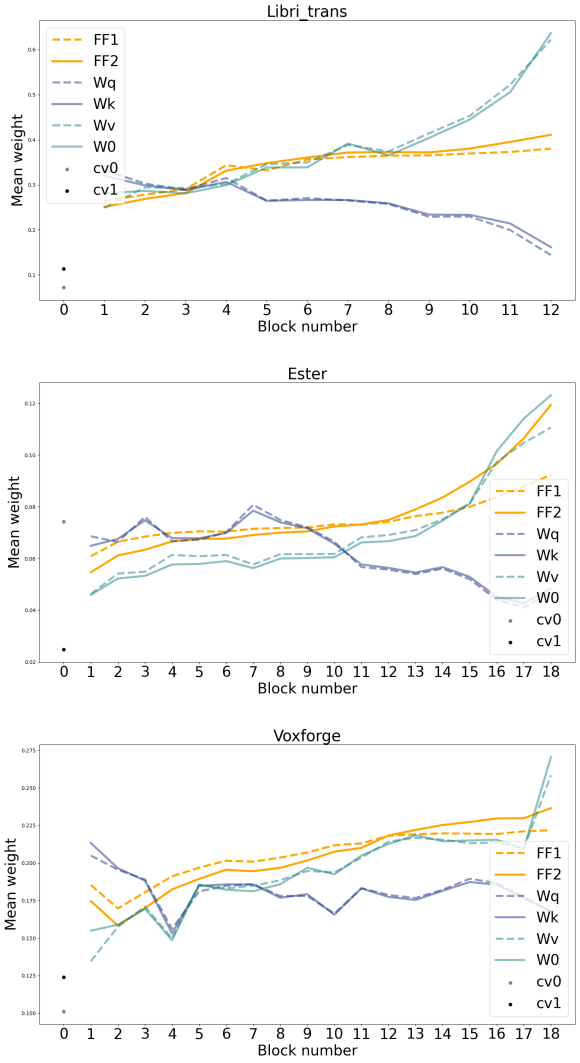



In all the models, the parameters of the feedforward layers are the most numerous exceeding 55% of the total number of parameters, those of the attention layers are above 22%, those of the convolution layers are lower than 3% and the rest of the layers represent less than 5%.

### B. Weight distribution

Weight pruning sets low-value weights to zero. Here we examine the weight values across the transformer encoder layers. The absolute values of the weights are averaged for each layer class, i.e., the convolution layers ($cv0$ and $cv1$), the feedforward layers $FF1$ and $FF2$, and the multi-head attention layers $W_q$, $W_k$, $W_v$, and $W_0$ and plotted in Fig. 3.

Fig. 3: Class weight distribution across the encoder blocks for the three models Libri-trans, Ester and Voxforge.




We notice that:

- the weights of the $FF1$ and $FF2$ layers are close to each other and have the same shape: they increase as the layer is deep.
- the curves of $W_k$ and $W_q$ are also very close. They decrease with the depth of the block except for Voxforge.
- regarding $W_v$ and $W_0$ layers, they are very close and increasing.

## VI. PRUNING EXPERIMENTS

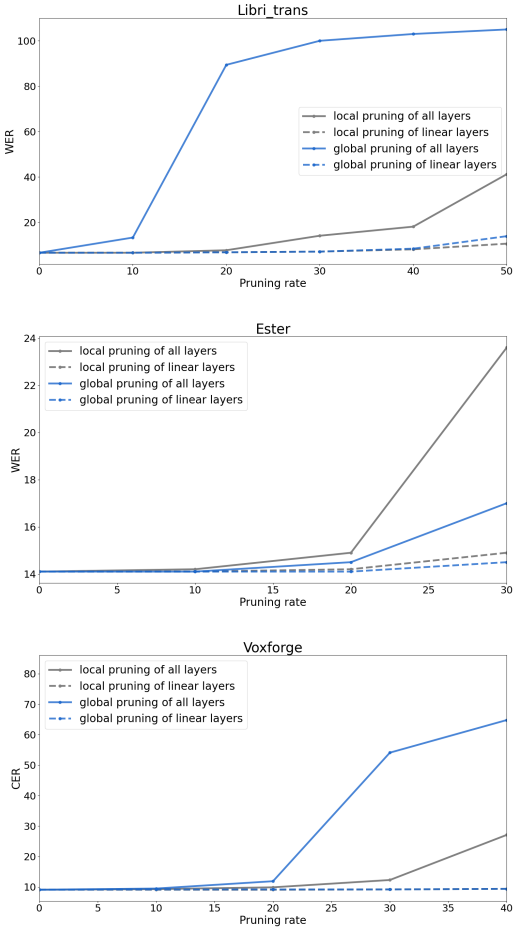
Pruning experiments are carried out on the Libri-trans, Ester and Voxforge transformer models in order to validate the

findings of Section V. We consider the main components of the transformer model, namely convolution layers (CONV), multi-head attention layers (MHA) and FeedForward (FF) layers.

### A. Impact of the convolution layers

To measure the impact of convolution layers on the model pruning, we performed experiments with and without convolution layers, i.e. pruning of all layers of the model or only linear layers that are MHA, FF, and input/output layers. Conventional methods of global and local pruning are employed.

Fig. 4: Error rate as a function of local and global pruning rates for Libri-trans, Ester and Voxforge models considering all layers or only linear layers.




Results are reported in Fig. 4. We notice that:

- Even if the convolution layers have very few weights, their global and local pruning increases the error rate. Especially for the Libri-trans model, when the pruning rate exceeds 10%, the WER increases rapidly.
- For the Ester model, local pruning is more sensitive to convolution layer pruning than global pruning. For Libri-trans and Voxforge, it is rather the opposite. Indeed, according to Fig. 3, the convolution layers of the Ester model have higher weight values.
- When only the linear layers are pruned, the global and local error rates are close to each other.

For these reasons, in the following pruning will be applied only to linear layers.

### B. FF layers behaviour

Now, let us examine the behavior of the feedforward layers. We consider global pruning and focus on the $FF1$ and $FF2$ layers. For different pruning rates, we report their sparsity through the encoder and decoder blocks respectively in Fig. 5 and Fig. 6. Each block contains an $FF1$ and an $FF2$ layer. The curves in continuous line represent the evolution of $FF1$, those in dashed line are that of $FF2$.

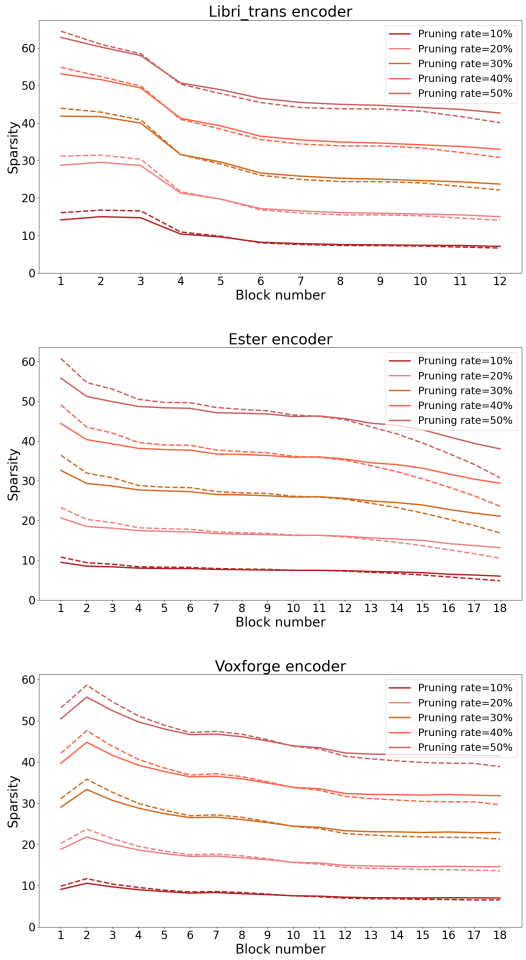



Fig. 5: $FF1$ and $FF2$ layers' sparsity for different global pruning rates through the encoder blocks. The curves in continuous line represent the evolution of $FF1$, those in dashed line are that of $FF2$.

In almost all cases, the feedforward layers $FF1$ and $FF2$ have decreasing sparsities, confirming the results of Fig. 3.

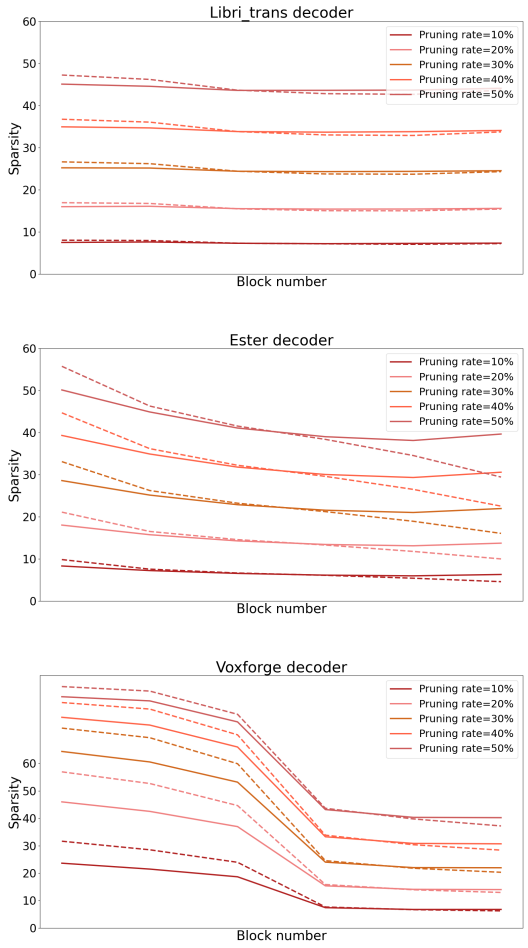



Fig. 6: $FF1$ and $FF2$ layers' sparsity for different global pruning rates through the decoder blocks. The curves in continuous line represent the evolution of $FF1$, those in dashed line are that of $FF2$.

Specifically, the mean weights of $FF1$ and $FF2$ are close and they decrease through the encoder (and decoder) blocks. Thus, the global pruning of the FF layers seems to be effective. However, we cannot say that the global pruning is optimal. Indeed, this method operates on all layers, whatever their nature or number of parameters and we have seen in the previous section that pruning convolution layers in our models increases the error rate drastically. As for local pruning, it can be argued that it is not adequate given the evolution of FF layer weights and the fact that local pruning assigns the same pruning rate to all layers.

### C. MHA layers behaviour

We are now interested in the behavior of the MHA layers namely the matrices $W_k$, $W_q$, $W_v$ and $W_0$. Considering the progression of weights plotted in Fig. 3, we propose to verify the conclusions of subsection V-B, i.e., that the values of the weights of matrices $W_k$ and $W_q$ are decreasing and that those of matrices $W_v$ and $W_0$ are increasing. To this end, we subdivided the MHA layers of the encoder into two parts: those of the first half of the encoder and those of its second half. Then we propose to apply different pruning rates to these layers depending on whether they are in the first or second half. More specifically, we try to verify that the pruning rate of the $W_k$ and $W_q$ layers must be low for the first layers and high for the deep layers. The opposite of this assumption must be true for the $W_v$ and $W_0$ layers, except for Voxforge.

To this end, we applied:

- a fixed pruning rate $x$ to all linear layers of the transformer model except the MHA layers of the encoder
- different pruning rates to the MHA layers of the first half of the encoder and its second half, so that the global pruning rate of the MHA layers is $x$.

We assume that $W_k$ and $W_q$ have equal pruning rates and similarly for $W_v$ and $W_0$. The variable pruning rates of the MHA layers are illustrated in the following table:

TABLE III: The pruning rates of the MHA layers of the encoder namely $W_k$, $W_q$, $W_v$ and $W_0$

| | First half layers | Second half of layers |
|---|---|---|
| $W_q$ and $W_k$ pruning rate | $am$ | $2x - am$ |
| $W_v$ and $W_0$ pruning rate | $2x - am$ | $am$ |

with $am$ a positive variable ($am < x$).

Fig 7 reports the error rates when fixing the overall pruning rate $x$ and varying the MHA layers rate using the variable $am$. When $x = am$, all MHA layers (in both the first and the second half of the encoder) have the same pruning rate. The corresponding error rate is highlighted on the curves by a large dot. In all curves, we compare the error rates for two parts: before the big point and after it. The global pruning error rate is indicated by a dotted line.

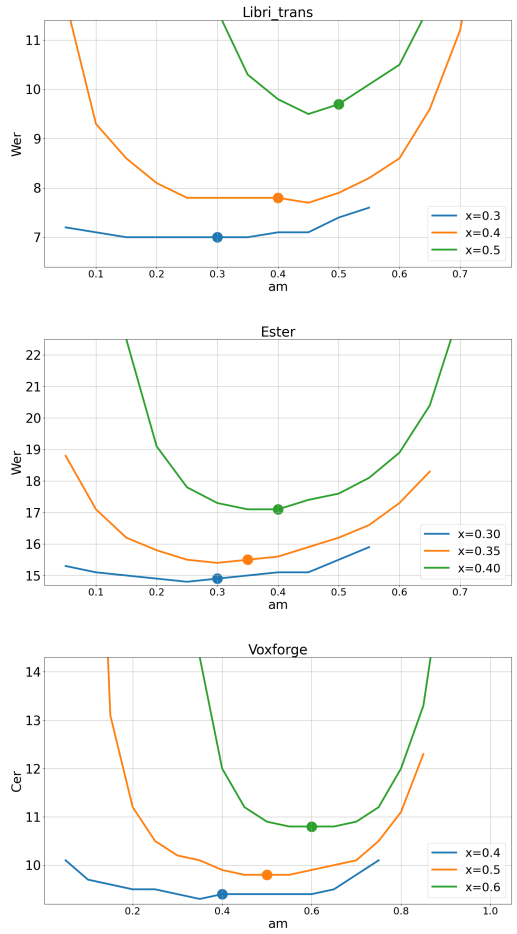
According to these curves, we draw some conclusions:

- For Libri-trans and Ester systems: The error rates to the left of the point are lower than those to its right. This corresponds to a low pruning rate of the first layers $W_k$ and $W_q$ and a high pruning rate of the deep layers $W_k$ and $W_q$. For the $W_v$ and $W_0$ layers, the opposite is true. These results support the assumptions made.
- Regarding Voxforge, it is often the right part of the curves that is lower. This means that it is better to apply high pruning rates to the first layers $W_k$ and $W_q$ as well as $W_v$ and $W_0$. Indeed, in Fig. 3 the weights of these layers are increasing. This again confirms the previous statements.
- The large point on each curve corresponds to the local pruning rate as all layers are pruned with the same rate. This point is often higher than the minimum, so local pruning is not efficient.

## VII. Conclusion and Perspectives

In this paper, we focus on weight pruning of transformer models in end-to-end speech recognition of three languages:

Fig. 7: Error rate as a function of the pruning rates of the layers $W_q$ (or $W_k$) for Libri-trans, Ester and Voxforge.




English, French and Italian. Standard weight pruning methods operate in a blind manner, with respect to the type of layers, the number of parameters or their behavior. Instead, we suggest a fine-grained analysis of the model layers. We find that pruning layers with few parameters is futile as it decreases performance without greatly reducing the model size. We also notice that the deeper the feedforward layer is, the more important it is and should be pruned less. Regarding the attention layers, their behavior changes with their position: The importance of the first two layers increases with depth, that of the last two decreases. Based on these findings, future work includes automatic analysis of model weights, followed by automatic custom pruning.

## 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] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.

[2] 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, p. 1135–1143.

[3] 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.

[4] L. B. Letaifa and M. I. Torres, "Perceptual borderline for balancing multi-class spontaneous emotional data," *IEEE Access*, vol. 9, 2021.

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

[6] M. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," in *International Conference on Learning Representations ICLR*, 2018.

[7] Z. Li, E. Wallace, S. Shen, K. Lin, K. Keutzer, D. Klein, and J. Gonzalez, "Train large, then compress: Rethinking model size for efficient training and inference of transformers," in *International Conference on Machine Learning ICML*, 2020, p. 2020.

[8] 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.

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

[10] 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 the International Conference on Acoustics, Speech, and Signal Processing*, 2019.

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

[12] 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.

[13] A. Bie, B. Venkitesh, J. Monteiro, M. A. Haidar, and M. Rezagholizadeh, "A simplified fully quantized transformer for end-to-end speech recognition," 2020.

[14] 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.

[15] L. B. Letaifa and J.-L. Rouas, "Transformer model compression for end-to-end speech recognition on mobile devices," in *The 30th European conference on signal processing EUSIPCO*, 2022.

[16] M. Gupta and P. Agrawal, "Compression of deep learning models for text: A survey," *ACM Trans. Knowl. Discov. Data*, 2020.

[17] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" in *Proceedings of Machine Learning and Systems*, I. Dhillon, D. Papailiopoulos, and V. Sze, Eds., 2020.

[18] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," in *International Conference on Learning Representations ICLR*, 2016.

[19] A. See, M.-T. Luong, and C. D. Manning, "Compression of neural machine translation models via pruning," in *SIGNLL Conference on Computational Natural Language Learning*, p. 291–301.

[20] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition." Proceedings ICASSP, 2018.

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

[22] 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.

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

[24] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *ASRU*, 2011.