On the Locality of Attention in Direct Speech Translation

Belen Alastruey*, **Javier Ferrando***, **Gerard I. Gállego** and **Marta R. Costa-jussà**
TALP Research Center, Universitat Politècnica de Catalunya, Barcelona
{belen.alastruey,javier.ferrando.monsonis
gerard.ion.gallego,marta.ruiz}@upc.edu

Abstract

Transformers have achieved state-of-the-art results across multiple NLP tasks. However, the self-attention mechanism complexity scales quadratically with the sequence length, creating an obstacle for tasks involving long sequences, like in the speech domain. In this paper, we discuss the usefulness of self-attention for Direct Speech Translation. First, we analyze the layer-wise token contributions in the self-attention of the encoder, unveiling local diagonal patterns. To prove that some attention weights are avoidable, we propose to substitute the standard self-attention with a local efficient one, setting the amount of context used based on the results of the analysis. With this approach, our model matches the baseline performance, and improves the efficiency by skipping the computation of those weights that standard attention discards.

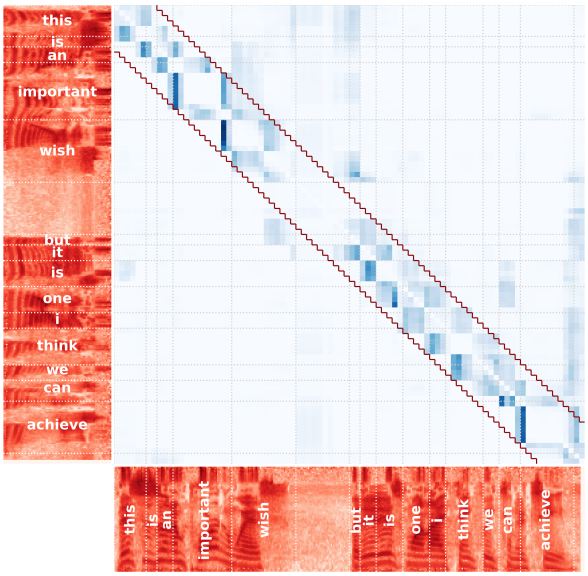

Figure 1: Spectrogram and contributions matrix[1] in Layer 11, after training for En-De ST. Tokens attend locally, creating a diagonal pattern. Highlighted is shown our proposed adaptive local attention window.

1 Introduction

Recently, Transformer-based models have gained popularity and have revolutionized Natural Language Processing (NLP) (Vaswani et al., 2017; Devlin et al., 2019; Brown et al., 2020). In the speech-to-text setting, the Transformer works with audio features like the mel-spectrogram (Dong et al., 2018; Di Gangi et al., 2019). These features provide longer input sequences compared to their raw text counterparts. This can be a problem when regarding complexity, since the Transformer's attention matrix computational cost is $O(n^2)$, where n is the sequence length. In speech, a common approach used to overcome this issue and reduce the input sequence length is to employ convolutional layers with stride before the Transformer encoder. However, even with the addition of convolutional layers, time and memory complexity is still an issue.

An active area of research has investigated ways to make the Transformer more efficient in tasks involving long documents, that exhibit the same problem as speech tasks (Tay et al., 2020). These models explore different techniques on how to avoid the computation of some attention weights, hence reducing the complexity of the self-attention layer. Some of these models, such as the Reformer (Kitaev et al., 2020) or the Routing Transformer (Roy et al., 2021), only compute attention weights on those queries and keys that are more related according to different clustering techniques. The authors of the Linformer (Wang et al., 2020b) state that the attention matrix is low-rank, so they project keys and values to reduce the size of the attention matrix. The Synthesizer (Tay et al., 2021) directly avoids computing token-to-token interac-

* Equal contribution.

[1] The main diagonal, which accounts for 65% of the total contributions, is hidden for visualization purposes.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Student Research Workshop, pages 402 - 412

tions by learning synthetic attention weights. The Longformer (Beltagy et al., 2020) and the Big Bird (Zaheer et al., 2020) modify the attention matrix with patterns such as local or random attentions. In this paper, we focus on local attention by using a sliding window centered on the diagonal of the attention matrix.

We build upon recent advances in the explainability of the Transformer to analyze the amount of context used by self-attention when dealing with speech features. Recent interpretability works have moved beyond raw attention weights as a measure of layer-wise input attributions and have integrated other modules in the self-attention, such as the norm of the vectors multiplying the attention weights (Kobayashi et al., 2020), the layer normalization, and the residual connection (Kobayashi et al., 2021). In the Automatic Speech Recognition (ASR) domain, the usefulness of the self-attention has been argued (Zhang et al., 2021; Shim et al., 2022), showing that its exposure to the full context might not be necessary, especially in the top layers. We carry out this analysis for Direct Speech Translation (ST) systems, which are capable of translating between languages from speech to text with a single model. The encoder of these systems needs to jointly perform acoustic and semantic modeling, while in ASR the latter is not that relevant (Liu et al., 2020). To the best of our knowledge, this is the first work that uses interpretability methods to understand how the Transformer's self-attention behaves in the Direct ST task.

In this work, we use the layer-wise contributions proposed by Kobayashi et al. (2021) to analyze the patterns of self-attention in Direct ST in En-De, En-Es and En-It tasks, unveiling their strong local nature. Consequently, using self-attention might not be entirely useful, but it is computationally costly. To verify this hypothesis, based on our analysis, we propose a new architecture designed to maximize the efficiency of the model while minimizing the information loss, and demonstrate no hinder in the model's performance in any of the three directions. We achieve this by substituting regular self-attention with local attention in those layers where the contributions are placed around the diagonal. Finally, we analyze the performance of the proposed model.

2 Speech-to-text Transformer

Recent works have attempted to adapt the Transformer to speech tasks (Di Gangi et al., 2019; Gulati et al., 2020). In the Direct ST domain, a usual approach is adding two convolutional layers with a stride of 2 before the Transformer (Wang et al., 2020a). By doing this, the sequence length is reduced to a fourth of the initial one. After the two convolutional layers, the speech-to-text Transformer (S2T Transformer) consists of a regular Transformer model, composed of 12 encoder layers and 6 decoder layers.

The main component of the Transformer is the multi-head attention mechanism, in particular, the self-attention is in charge of mixing contextual information. Given a sequence of token representations $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, each of the H heads projects these vectors to queries $\mathbf{Q}^h \in \mathbb{R}^{N \times d_h}$, keys $\mathbf{K}^h \in \mathbb{R}^{N \times d_h}$ and values $\mathbf{V}^h \in \mathbb{R}^{N \times d_h}$, with head dimension $d_h = d/H$, where d is the model embedding dimensionality. The self-attention attention (SA) computes:

$$\text{SA}(\mathbf{Q}^h, \mathbf{K}^h, \mathbf{V}^h) = \sum_h^H \mathbf{A}^h \mathbf{V}^h \mathbf{W}_O^h + \mathbf{b}_O \quad (1)$$

Where $\mathbf{W}_O^h \in \mathbb{R}^{d_h \times d}$ and $\mathbf{b}_O \in \mathbb{R}^d$ are learnable parameters, and

$$\mathbf{A}^h = \text{softmax}\left(\frac{\mathbf{Q}^h (\mathbf{K}^h)^T}{\sqrt{d_h}}\right) \quad (2)$$

Training details. We reproduce the S2T Transformer training with FAIRSEQ (Ott et al., 2019; Wang et al., 2020a). The training procedure consists of two phases. First, we pre-train the model in the ASR setting (Bérard et al., 2018). Then, we substitute the decoder with a randomly initialized one, and both are finally trained in the ST task (see Appendix C for more details on the hyperparameters). For the trainings, we use the MUST-C English-German, English-Spanish and English-Italian subsets (Cattoni et al., 2021).

3 Model Analysis

In this section, we present the analysis of the encoder self-attention in the S2T Transformer.

Interpretability method. Kobayashi et al. (2021) propose an interpretability method that measures the impact of each layer input, i.e token representations ($\mathbf{x}_j$), to the output of the layer,

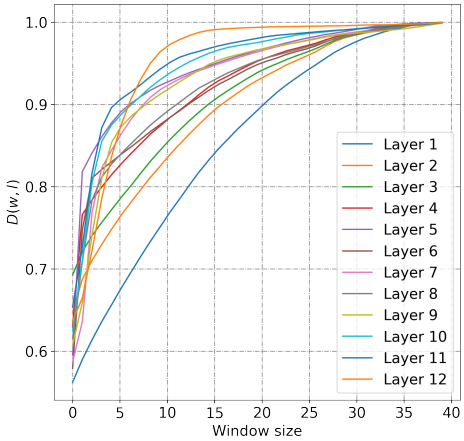

Figure 2: Contribution diagonality $D(w, l)$ after ST training, for a single En-De example. The greater the area under the curve (CCD), the higher the diagonality.

considering also the layer normalization and the residual connection. They provide the formulation for the attention block of the original Transformer architecture, which has layer normalization on top of the self-attention module. In this work, we give an adaptation to the group of models that normalize before the multi-head attention (Pre-LN), such as the S2T Transformer. The complete chain of computations in the Pre-LN attention block can be reformulated as a simple expression of the layer inputs:

$$\hat{\mathbf{x}}_i = \sum_j^N \sum_h^H \mathbf{A}_{i,j}^h \mathrm{LN}(\mathbf{x}_j)\mathbf{W}_V^h \mathbf{W}_O^h + \mathbf{b}^O + \mathbf{x}_i \quad (3)$$

We can now express the attention block output as a sum of transformed input vectors ($F_i(\mathbf{x}_j)$):

$$\hat{\mathbf{x}}_i = \sum_j^N F_i(\mathbf{x}_j) + \mathbf{b}^O \quad (4)$$

Where $F_i(\mathbf{x}_j)$ is defined as:

$$F_i(\mathbf{x}_j) = \begin{cases} \sum_h^H \mathbf{A}_{i,j}^h \mathrm{LN}(\mathbf{x}_j)\mathbf{W}_V^h \mathbf{W}_O^h & \text{if } j \neq i \\ \sum_h^H \mathbf{A}_{i,j}^h \mathrm{LN}(\mathbf{x}_j)\mathbf{W}_V^h \mathbf{W}_O^h + \mathbf{x}_i & \text{if } j = i \end{cases}$$

Kobayashi et al. (2021) measure the contribution $C_{i,j}$ of each input vector $\mathbf{x}_j$ to the layer output $\hat{\mathbf{x}}_i$ with the Euclidean norm of the transformed vector:

$$C_{i,j} = \|F_i(\mathbf{x}_j)\| \quad (5)$$

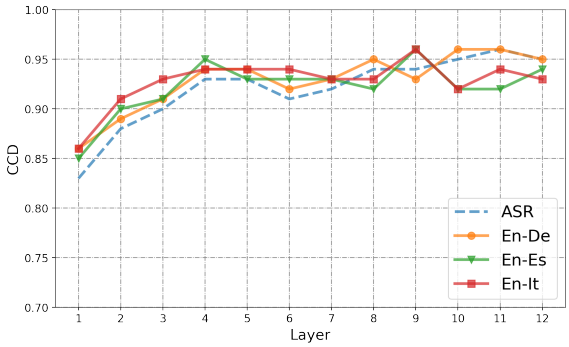

Figure 3: Average cumulative contribution diagonality (CCD) score across layers, over 100 samples. Results shown for models trained in ASR (dashed line) and ST (solid line).

Layer-wise analysis. We analyze the contribution scores obtained with Eq. 5 from the encoder layers in both ASR (pre-training) and ST tasks. From the results shown in Figure 4 (see also Appendix E) we observe that most layers' contributions are dense around the diagonal. To measure the degree of diagonality in the contribution matrices at each layer l, we build upon the attention diagonality proposed by Shim et al. (2022), originally defined with attention weights and proportions of the sequence length. We reformulate it with the obtained contributions, and token ranges w (see Appendix A for more details on the differences):

$$D(w, l) = \frac{1}{N} \sum_i \sum_j C_{i,j}^l \quad (6)$$

where $j \in [\max(1, i - \lfloor \frac{1}{2} w \rfloor), \min(N, i + \lfloor \frac{1}{2} w \rfloor]$, $i \in [1, N]$. $D(w, l)$ computes the average of the contributions restricted by the diagonal window range w. In order to measure how fast the contribution density increases over the window length, we calculate the cumulative contribution diagonality (CCD), that corresponds to the area under the curve of the accumulated $D(w, l)$ within the range[2] $w \in [1, 2N]$. That is, we approximate the integral of $D(l, d)$ along the distance d, but for the discrete variable w (Figure 2).

In Figure 3 we show the CCD results for ASR and ST across layers, where we can observe a strong diagonal pattern. We can see that, surprisingly, CCD is very similar in both tasks. This contradicts the belief that, because of the need for deeper semantic processing when translating, ST

[2]Note that a window of size w contains $\lfloor \frac{1}{2} w \rfloor$ tokens on each side of the main diagonal, so $w = 1$ represents the main diagonal and $w = 2N$ every possible diagonal.

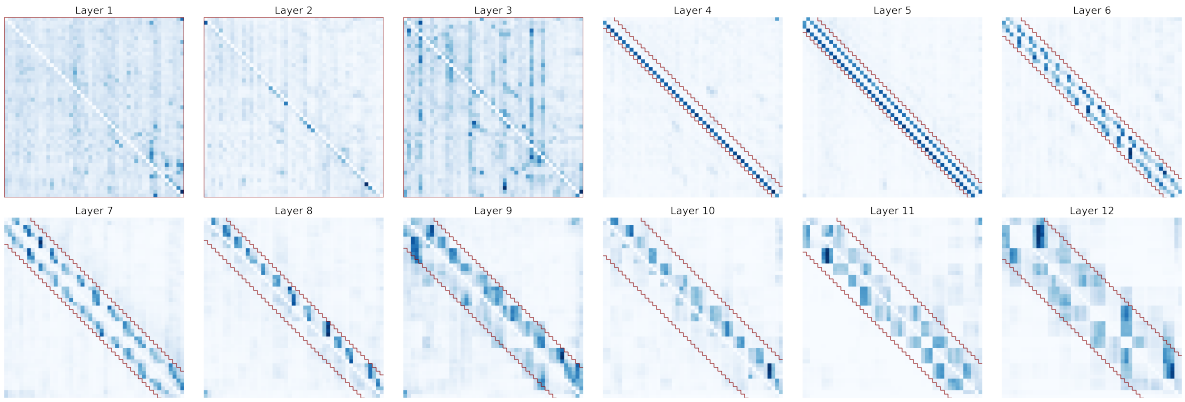

Figure 4: Contribution matrices[3] of encoder layers on a sample after training for En-De ST. Windows used in the efficient architecture highlighted.

needs more context than ASR. Furthermore, we see different behaviors along the encoder, and a trend towards uniformly distributed contributions in the first layers.

Moreover, in Figure 4, we can see differences between those layers that show local patterns. Layers 4, 5 and 6 attend to close context. Instead, those layers at the end of the encoder, as 11 or 12, need larger context, and we can see how the contributions create patterns corresponding to words in the spectrogram, enabling us to see interactions between them (Figure 1). Additionally, we see that contribution matrices reveal silences in the speech sequence. However, we believe that further research is needed to fully understand the meaning of these patterns.

4 Efficient Speech-to-text Transformer

From the previous analysis, we hypothesize that suitable local attention patterns may potentially avoid the computation of unused attention scores. Note that if a token does not contribute to the output of a layer, its attention score can be canceled. Our objective is to maximize the efficiency of the model while minimizing the performance drop.

Window size selection. CCD could serve as a starting point to obtain optimal window lengths. However, it requires predefining the amount of total contribution required inside the window, which makes it fragile to properly detect local patterns. On one hand, it can be too sensitive to a strong main diagonal. On the other, it may overestimate random distant contributions. We propose an alternative based on the average contribution of every sub/superdiagonal (Alg. 1). Starting from the main diagonal $C_{diag}[0]$, it keeps adding tokens to the window length until it finds $N \cdot 10\%$ consecutive sub/superdiagonals below the t threshold[4]. We repeat this procedure with a random set of 400 sentences, and we compute each layer's window length mean (μ^l) and standard deviation (σ^l). To ensure that most significant contributions are considered, we define as the optimal window size (w^l) the result of the operation[5] $w^l = \lceil \mu^l + \sigma^l \rceil$. The results obtained are similar in every language pair (Table 1 and Appendix D). In Figure 4, we can see how w contains most of the relevant contributions for En-De ST (see more examples in Appendix E).

Algorithm 1: Window size selection

Input:
C^l: contribution matrix, N: number of tokens,
t: min diagonal contribution threshold
Output:
w^l: optimal window size

$counter \leftarrow 0$
$w^l \leftarrow 0$
while $counter < (N/10)$ **do**
  for $i \leftarrow 0, N$ **do**
    if $mean(C^l_{diag}[i]) > t$ ***or*** $mean(C^l_{diag}[-i]) > t$
    then
      $w^l \leftarrow 2i + 1$
      $counter \leftarrow 0$
    else
      $counter \leftarrow counter + 1$

[3] The main diagonal, which accounts for around 65% of the total contributions in each layer, is hidden for visualization purposes.

[4] Hyperparameter that defines the minimum average value of the sub/superdiagonals to be considered. We choose 0.01 after empirical study.

[5] If w^l is even, we set $w^l = w^l + 1$ so that it is odd and hence the window can be centered around the diagonal.

Layer	$\mu \pm \sigma$	w	CL
1 *	3.41 ± 13.15	*17*	*0.35 ± 0.07*
2 *	1.18 ± 3.45	*5*	*0.32 ± 0.04*
3 *	0.51 ± 1.56	*3*	*0.30 ± 0.04*
4	2.25 ± 1.30	5	0.23 ± 0.04
5	4.03 ± 0.28	5	0.17 ± 0.03
6	7.03 ± 1.03	9	0.23 ± 0.04
7	11.37 ± 1.13	13	0.18 ± 0.04
8	7.94 ± 1.16	11	0.18 ± 0.04
9	12.56 ± 1.85	15	0.19 ± 0.05
10	16.47 ± 2.40	19	0.13 ± 0.05
11	13.28 ± 1.90	17	0.13 ± 0.04
12	16.28 ± 3.86	21	0.16 ± 0.05

Table 1: Optimal window size study in En-De ST. (*) For the first three layers, we use standard self-attention.

	En-De	**En-Es**	**En-It**
Baseline	22.53 ± 0.15	27.49 ± 0.22	22.98 ± 0.15
Ours	22.49 ± 0.11	27.46 ± 0.12	22.97 ± 0.27

Table 2: BLEU obtained on the Speech Translation task (mean ± std after training with 5 different seeds).

Contribution loss. We can now calculate the percentage of the total contributions that are left outside the window. This allows us to discover the amount of contribution that is lost because of the use of local attention. To do so, we employ Eq. 6, but since we are interested in the contributions outside each window w^l, we define $CL(l, w^l) = 1 - D(l, w^l)$.

Proposed architecture. From the previous results, we see that the first three layers are the ones with the weakest local pattern (See Figure 4). In these layers, $CL(l, w^l)$ is large, and CCD (Figure 2) shows smaller areas. For these reasons, we believe that using the entire self-attention in the first three layers is necessary. In the following layers, we use local attention with window size w^l. Our proposed architecture is an efficient adaptation of the S2T Transformer, and therefore it is exactly equal with exception of the self-attention layers (detailed architectures in Section 2 and Appendix B).

Experiments. Finally, we train our model under the same specifications and dataset as the baseline (see Section 2 for details on the dataset, and Appendix C for the training hyperparameters).

As we see in Table 2, our model matches the performance of the S2T Transformer in every analyzed language pair. However, we achieve it while reducing the complexity in most layers from $O(n^2)$ to $O(n \cdot w^l)$. This difference can be highly significant, considering the usual length of speech sequences and the size of the windows used. In particular, w^l goes from 5 to 25 tokens between the different languages. However, the average length of an input sequence in studied splits of the MUST-C dataset after the two convolutional layers used in the S2T Transformer, is 166 tokens, even reaching a maximum of 1052.

5 Conclusions

Transformer-based models are the current state-of-the-art in many different fields. However, the quadratic complexity of the self-attention module usually hinders the usefulness of the model in real-life applications. This problem worsens when working with long sequences, as is the case with speech. In this paper, we have questioned the need of computing all attention weights in ST. We have analyzed the contribution matrices, and we have seen that, in many layers, the relevant scores are placed in a diagonal pattern. Therefore, we have hypothesized that these weights do not need to be calculated. To verify our hypothesis, we have trained a model that substitutes regular self-attention with local attention, with a suitable window size for each layer. We have seen that as we expected, the results are almost equal to the ones obtained with the baseline model, but the complexity has been lowered significantly.

Regarding interpretability, we have found how the Transformer establishes connections between words in speech sequences. Furthermore, we have seen that, in contrast to what was expected, diagonality scores are similar in both ST and ASR tasks, meaning that they use the same amount of context.

6 Acknowledgments

This work was partially funded by the project ADAVOICE, PID2019-107579RB-I00 / AEI / 10.13039/501100011033, and the UPC INIREC scholarship nº3522. We would like to thank Ioannis Tsiamas and Carlos Escolano for their support and advice, and the anonymous reviewers for their useful comments.

7 Ethical Considerations

This work analyzes the inner workings of a particular architecture in Direct Speech Translation. Based on the analysis, we propose a more efficient model, that maintains the baseline performance.

Our proposed solution can help reduce the ecological footprint of Speech Translation systems based on the Transformer architecture. We believe this work has no direct negative social influences. However, we should underline that the dataset used in this paper consists of high-resource languages such as English, German, Spanish, and Italian. Although the interpretability method does not depend on specific languages, there may be differences in the degree of efficiency that can be achieved when experimenting with other languages.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6224–6228.

Roldano Cattoni, Mattia Antonino Di Gangi, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2021. Must-c: A multilingual corpus for end-to-end speech translation. *Computer Speech & Language*, 66:101155.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia Antonino Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting Transformer to End-to-End Spoken Language Translation. In *Proc. Interspeech 2019*, pages 1133–1137.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2021. Incorporating Residual and Normalization Layers into Analysis of Masked Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4547–4568, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yuchen Liu, Junnan Zhu, Jiajun Zhang, and Chengqing Zong. 2020. Bridging the modality gap for speech-to-text translation. *ArXiv*, abs/2010.14920.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Kyuhong Shim, Jungwook Choi, and Wonyong Sung. 2022. Understanding the role of self attention for efficient speech recognition. In *International Conference on Learning Representations*.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2021. Synthesizer: Rethinking self-attention in transformer models. In *International Conference on Machine Learning*. PMLR.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *ArXiv*, abs/2009.06732.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020a. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 33–39, Suzhou, China. Association for Computational Linguistics.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020b. Linformer: Self-attention with linear complexity. *ArXiv*, abs/2006.04768.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.

Shucong Zhang, Erfan Loweimi, Peter Bell, and Steve Renals. 2021. On the usefulness of self-attention for automatic speech recognition with transformers. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 89–96.

A Cumulative Attention Diagonality (CAD)

Shim et al. (2022) propose the cumulative attention diagonality (CAD) as the integral of the attention diagonality $D(r, l)$ along the variable r, which defines the window length as a proportion of the sequence length:

$$CAD^l = \int_{r=0}^{r=1} D(r, l)\ dr$$

where $D(r, l)$ is defined over the attention weight matrix A^l:

$$D(r, l) = \frac{1}{N} \sum_{i=1}^{N} \sum_{\substack{j=\max(1,\\ i-r(N-1)}}^{\substack{\min(N,\\ i+r(N-1))}} A^l_{i,j}$$

To approximate the result of the integral, Shim et al. (2022) use the Trapezoidal Rule with the discretized variable $\hat{r} \approx r$.

$$\int_{r=0}^{r=1} D(r, l)\ dr \approx \sum_{\hat{r}=0}^{\hat{r}=1} \frac{D(\hat{r}, l) + D(\hat{r}+1, l)}{2}$$

For each step in the summation, the window range around the diagonal increases $2r(N-1)$, which may lead to different increments based on the sentence length. For instance, for a sentence with $N = 11$, in a 0.1 increase of $\hat{r}$, the window size range increases by 2. However, with $N = 101$ we get an increment of 20. For this reason, we redefine the diagonality measures with token-wise increments.

B Architecture Details

Both our efficient model and the S2T Transformer (Wang et al., 2020a) share the same architecture, with the exception of the self-attention modules. The models consist 12 encoder layers and 6 decoder layers with sinusoidal positional encodings. In the encoder and decoder we use 4 attention heads, an embedding dimension of 256, and of 2048 in the FFN layers. We use a dropout probability of 0.1 in both the attention weights and FFN activations. We use ReLU as the activation function.

Regarding the convolutional layer applied to reduce sequence length, it consists of a 1D convolutional layer, with a kernel of size 5, a stride of 2, and with the same number of output channels than input channels.

C Training Hyperparameters

To ensure a reliable comparison, we performed all ASR and ST experiments under the same conditions and hyperparameters. In ASR training we fixed a maximum of 40000 tokens per batch. We used Adam optimizer and a learning rate of $1 \cdot 10^{-3}$ with an inverse square root scheduler. We applied a warm-up for the first 10000 updates. We clipped the gradient to 10 to avoid exploding gradients. We used label smoothed cross-entropy as a loss function, with a smoothing factor of 0.1. We used an update frequency of 8 on a single GPU. We set a maximum of 50000 updates for every training. In ST training, we use the same hyperparameters as for ASR, but we use a learning rate of $2 \cdot 10^{-3}$. We conducted the training of all our experiments using NVIDIA GeForce RTX 2080 Ti GPU.

D Optimal window analysis in En-Es and En-It ST

Layer	$\mu \pm \sigma$	w	CL
1 *	4.68 ± 14.77	*21*	*0.39 ± 0.08*
2 *	3.21 ± 6.17	*11*	*0.29 ± 0.04*
3 *	0.99 ± 3.6	*5*	*0.28 ± 0.04*
4	2.58 ± 1.96	5	0.2 ± 0.02
5	4.52 ± 2.38	7	0.24 ± 0.04
6	15.88 ± 2.92	19	0.21 ± 0.06
7	11.32 ± 1.91	15	0.16 ± 0.03
8	9.52 ± 2.5	13	0.22 ± 0.05
9	14.96 ± 1.78	17	0.07 ± 0.05
10	15.94 ± 3.0	19	0.19 ± 0.05
11	13.83 ± 3.66	19	0.21 ± 0.05
12	20.38 ± 3.42	25	0.1 ± 0.05

Table 3: Optimal window size study in En-Es ST. (*) For the first three layers, we use standard self-attention.

Layer	$\mu \pm \sigma$	w	CL
1 *	6.16 ± 17.57	*25*	*0.34 ± 0.09*
2 *	2.56 ± 7.47	*11*	*0.29 ± 0.05*
3 *	2.44 ± 2.84	*7*	*0.27 ± 0.05*
4	4.08 ± 0.65	5	0.19 ± 0.03
5	14.05 ± 2.08	17	0.15 ± 0.03
6	10.82 ± 1.31	13	0.18 ± 0.04
7	7.37 ± 4.54	13	0.23 ± 0.05
8	8.62 ± 2.18	11	0.22 ± 0.04
9	12.49 ± 1.65	15	0.09 ± 0.03
10	16.06 ± 3.80	21	0.17 ± 0.04
11	18.15 ± 3.20	23	0.11 ± 0.05
12	17.34 ± 4.83	23	0.15 ± 0.05

Table 4: Optimal window size study in En-It ST. (*) For the first three layers, we use standard self-attention.

E Contribution Matrices

Below, we show more examples of contribution matrices for the different languages that have been studied. Note that, although in some cases a diagonal a pattern appears in the first three layers, the diagonality score is still low. Local diagonal patterns are not strictly related to high diagonality, since contributions outside the pattern might be uniformly distributed, and thus difficult to observe in the heatmap. For this reason, contribution matrices can be misleading, and we focus on the use of CL scores to determine which layers should use full attention.

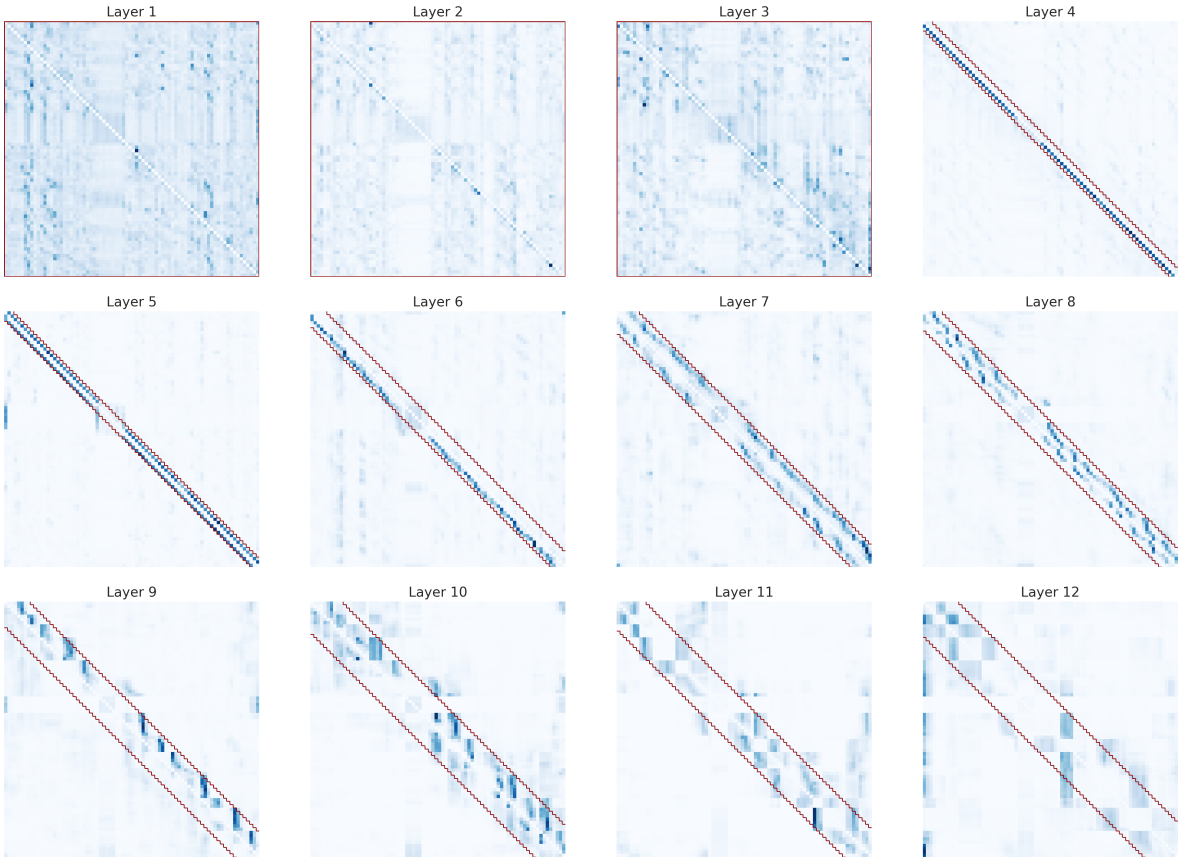

Figure 5: Contribution matrices for a sample after En-De ST training.

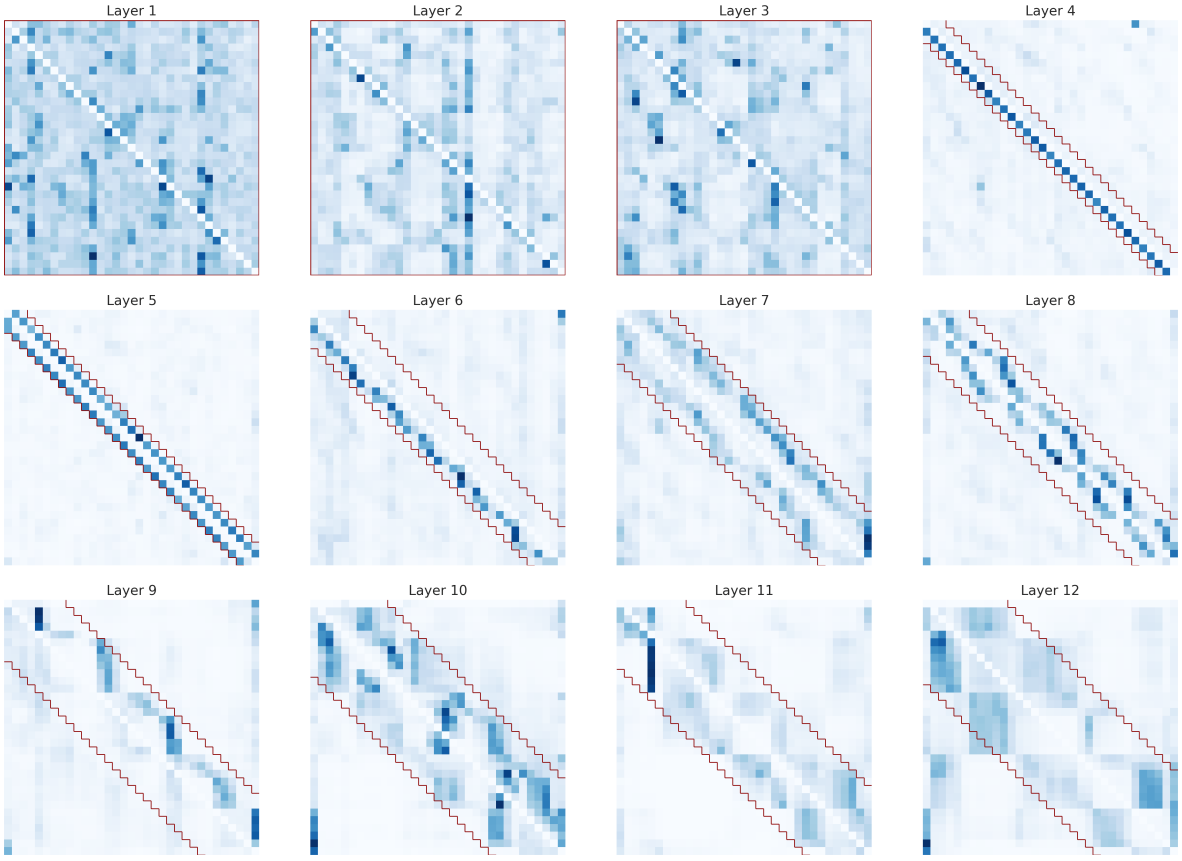

Figure 6: Contribution matrices for a sample after En-De ST training.

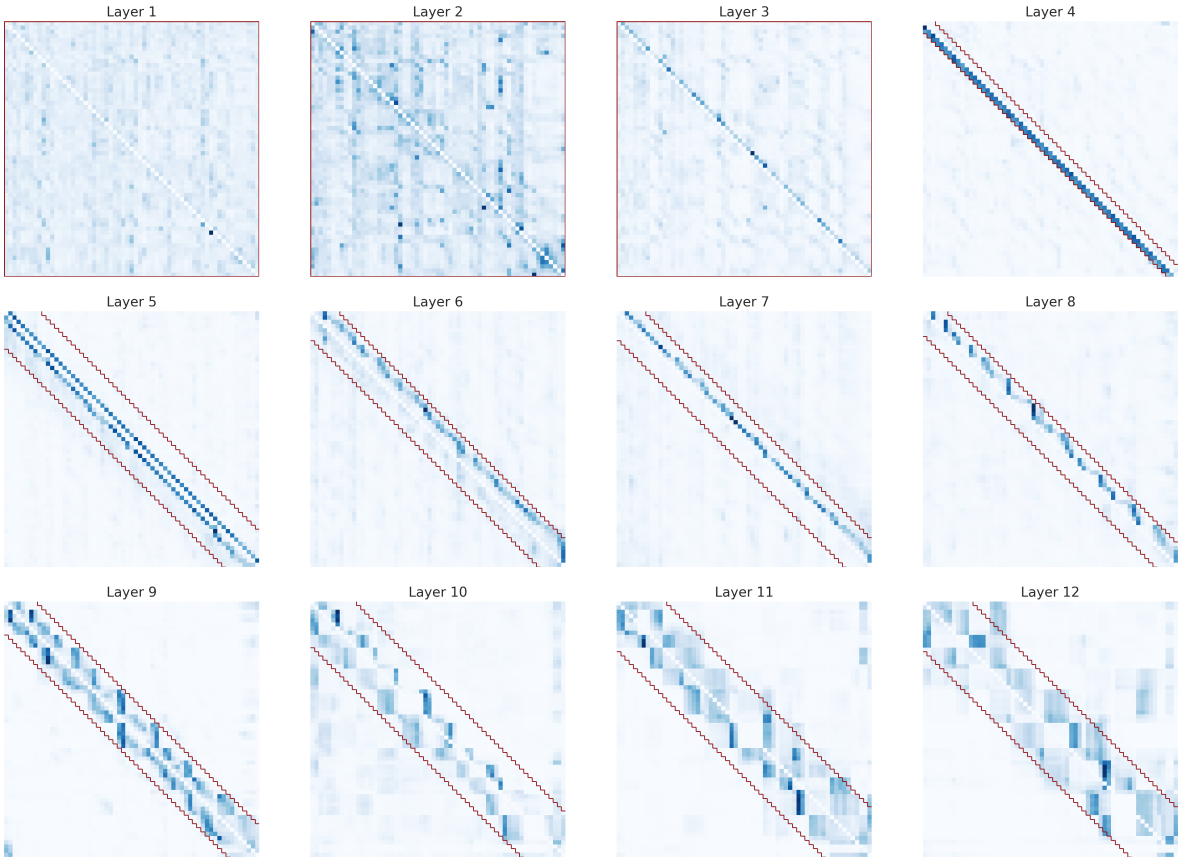

Figure 7: Contribution matrices for a sample after En-It ST training.

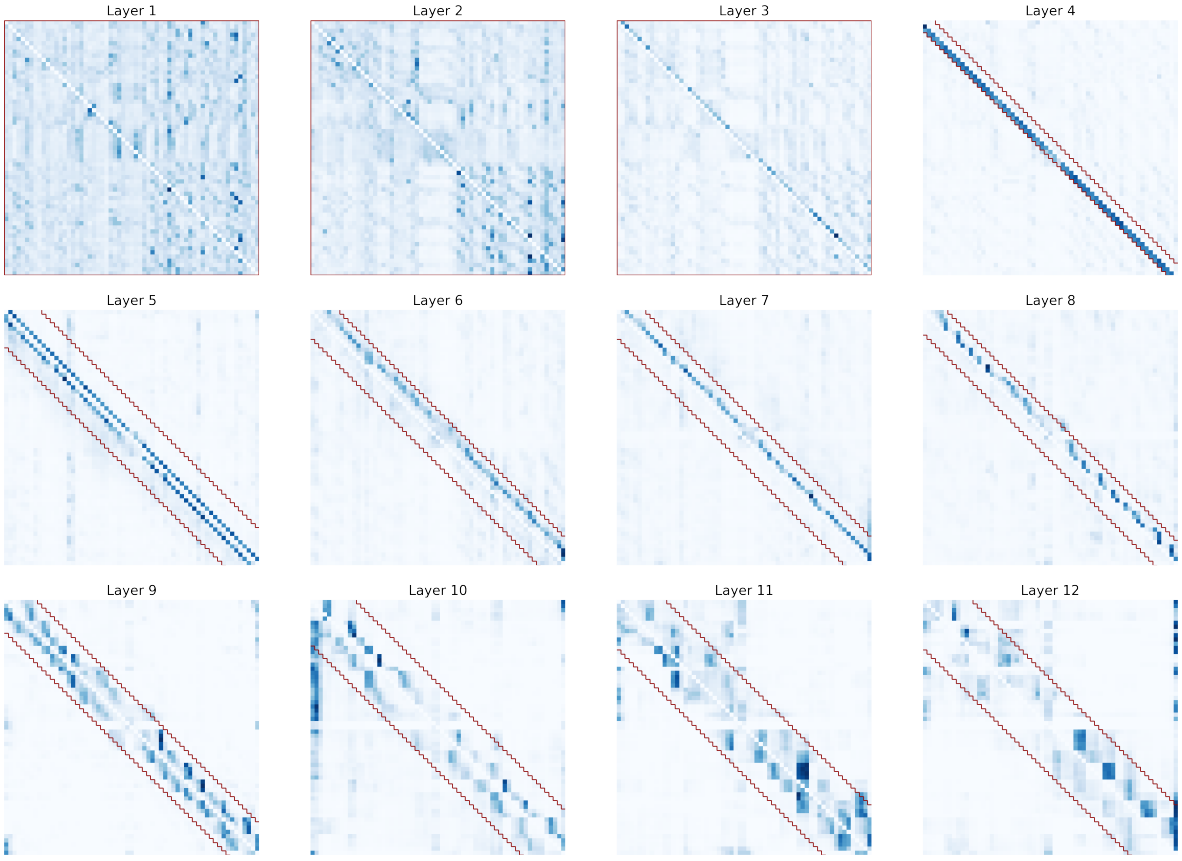

Figure 8: Contribution matrices for a sample after En-It ST training.

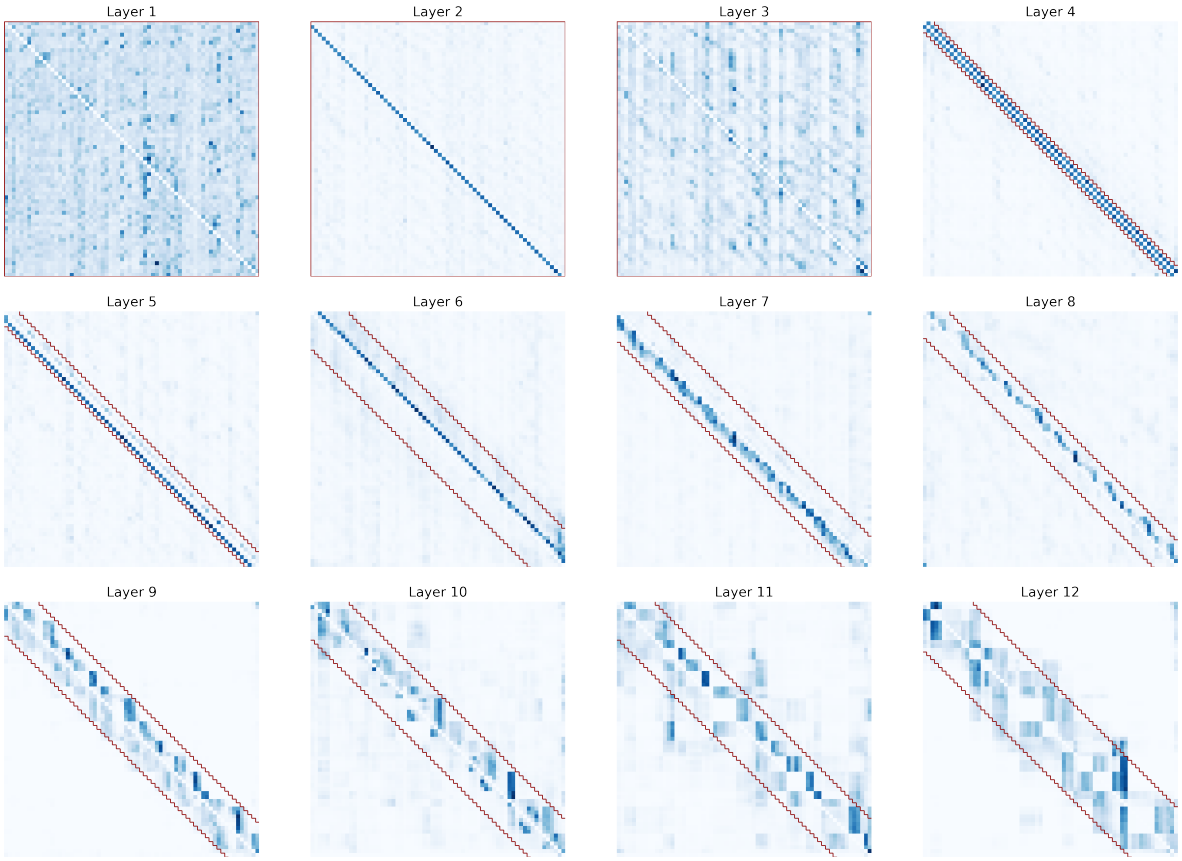

Figure 9: Contribution matrices for a sample after En-Es ST training.

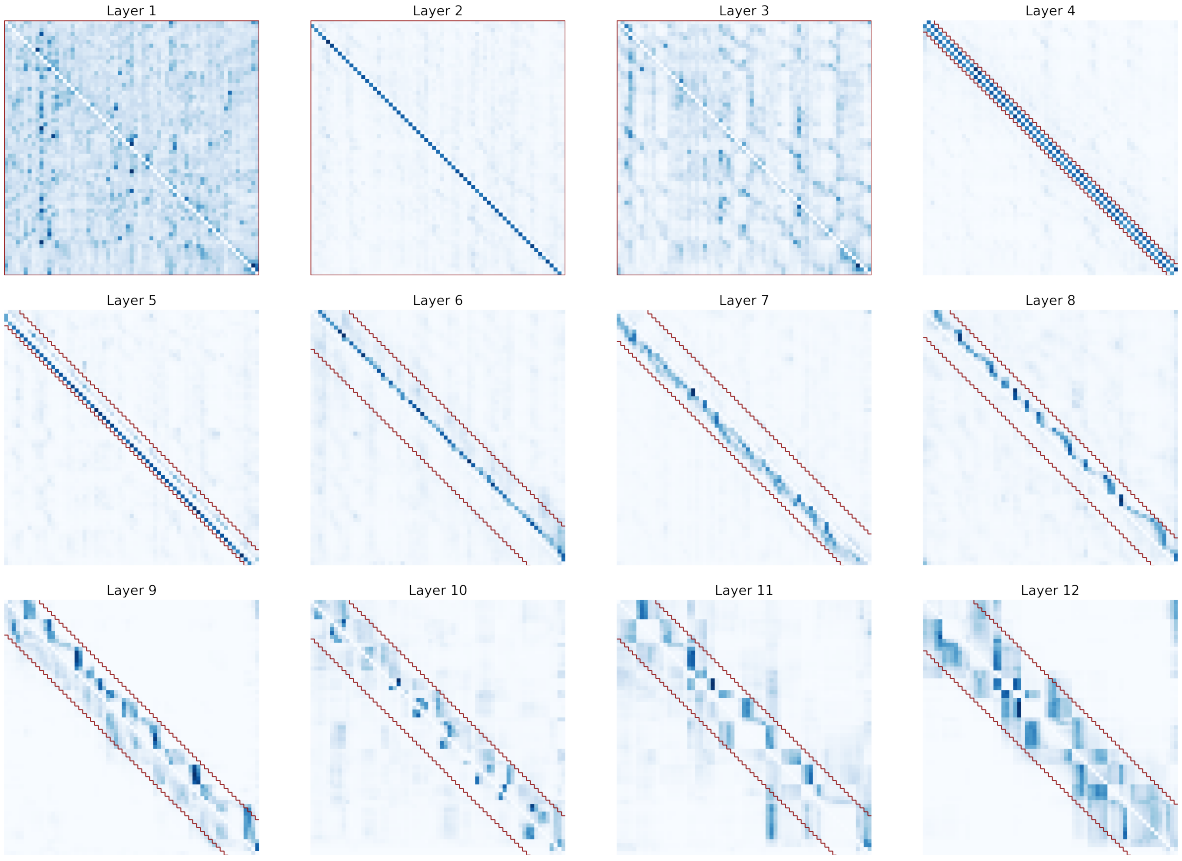

Figure 10: Contribution matrices for a sample after En-Es ST training.