ACL-IJCNLP 2021

The Case for Translation-Invariant Self-Attention in Transformer-Based Language Models

Ulme Wennberg

Gustav Eje Henter

Contributions

- Analyze the positional processing in state-of-the-art language models
- Leverage this analysis to propose first positional information processing method that simultaneously satisfies a number of key design criteria

Positional dependencies in transformer models

Do we need to model positional dependencies in transformer models?

How do we model positional dependencies in transformer models?

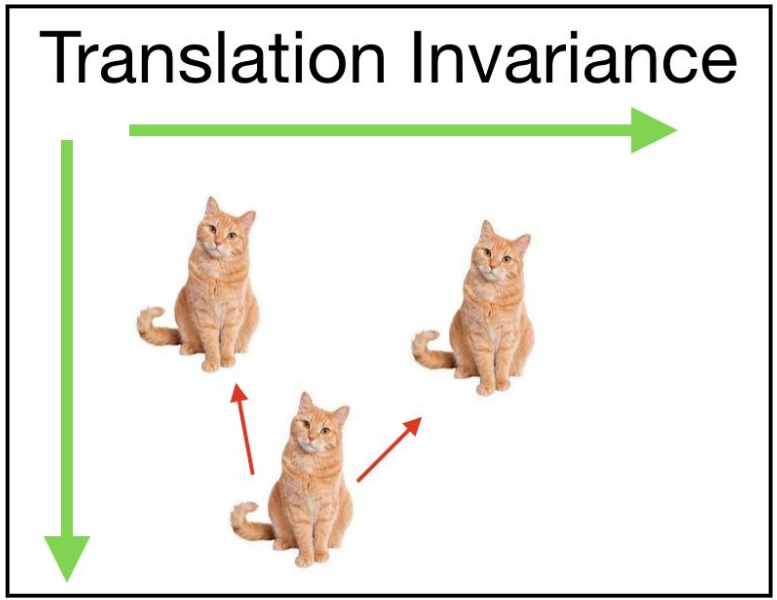
Where do we model positional dependencies in transformer models?

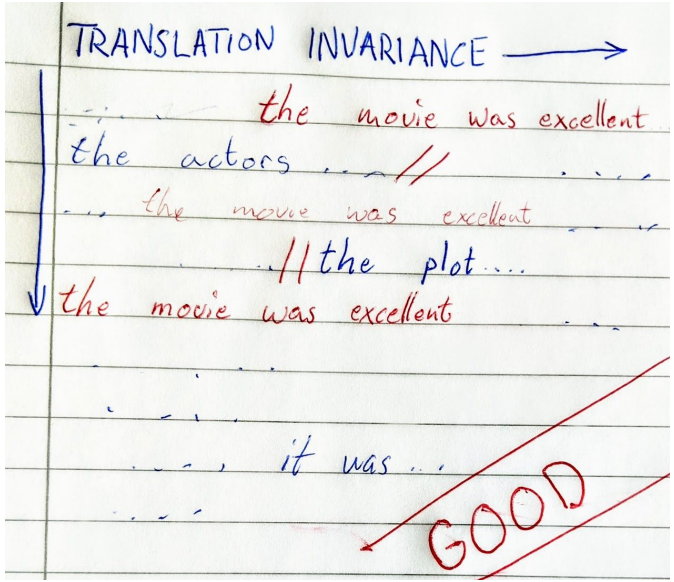
Design criteria of positional dependency modeling in transformer models

Method	Induct-ive?	Data-driven?	Parameter efficient?	Translation invariant?	Parallel -izable?	Interpret-able?
Sinusoidal position embedding (Vaswani et al., 2017)	✓	✗	✓	✗	✓	✗
Absolute position embedding (Devlin et al., 2019)	✗	✓	✗	✗	✓	✗
Relative position embedding (Shaw et al., 2018)	✗	✓	✓	✓	✗	✗
T5 (Raffel et al., 2020)	✗	✓	✓	✓	✓	✓
Flow-based (Liu et al., 2020)	✓	✓	✓	✗	✗	✗
Synthesizer (Tay et al., 2020)	✗	✓	✓	✗	✓	✗
Untied positional scoring (Ke et al., 2021)	✗	✓	✗	✗	✓	✗
Rotary position embedding (Su et al., 2021)	✓	✗	✓	✓	✓	✗
Translation-invariant self-attention (proposed)	✓	✓	✓	✓	✓	✓

Table 1: Characteristics of position-representation approaches for different language-modeling architectures.

What does translation invariance look like in text?

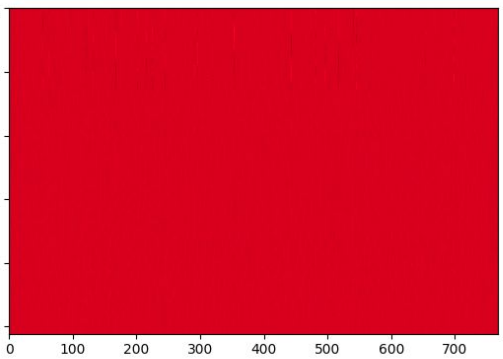




Position embeddings are not interpretable

Each row in the position embedding matrix Ep represents one position embedding vector

$E_p =$

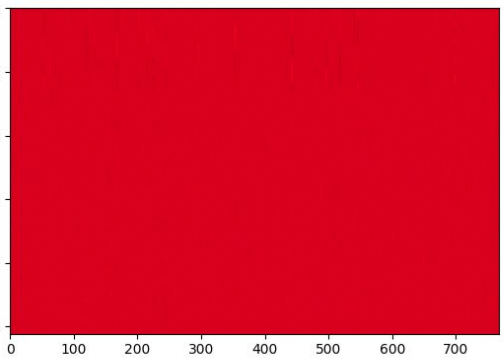


There are 512 embeddings,
each with 768-dimensions
in BERT base

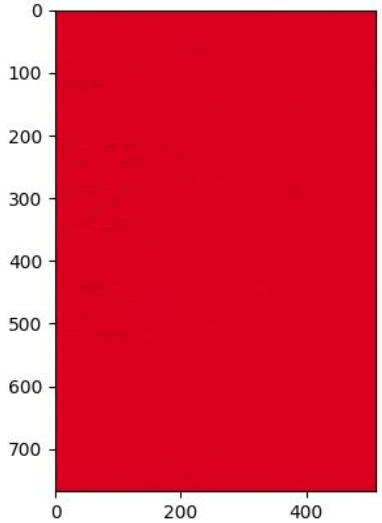
Position embeddings are not interpretable

Each row in the position embedding matrix Ep represents one position embedding vector

E_p =



E_p ’=

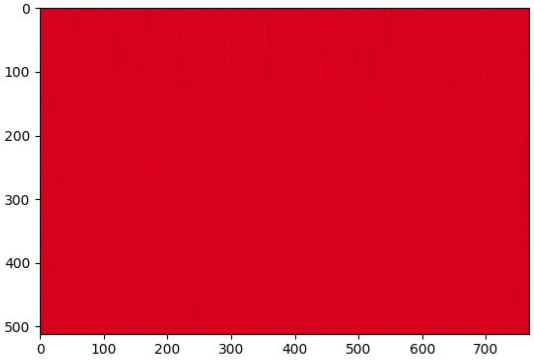


There are 512 embeddings, each with 768-dimensions in BERT base

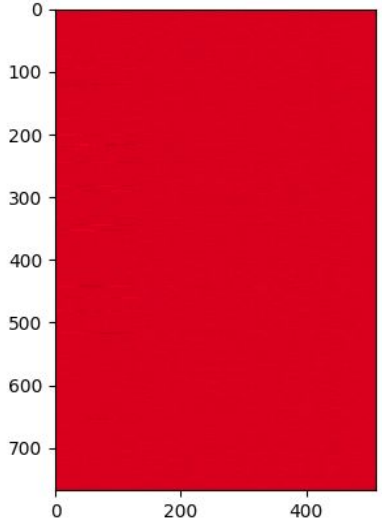
Position embeddings are not interpretable

Each row in the position embedding matrix Ep represents one position embedding vector

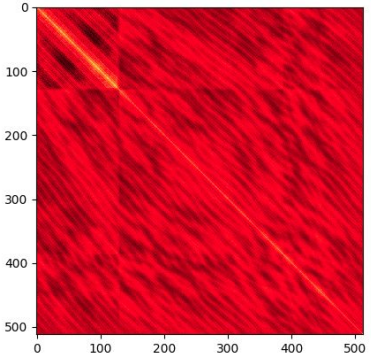
E_p =



E_p'=



$E_p * E_p$'=

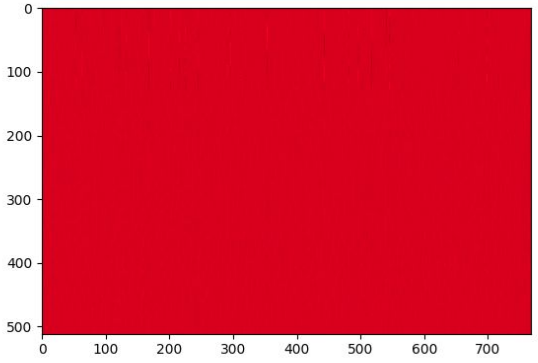


There are 512 embeddings, each with 768-dimensions in BERT base

Position embeddings are not interpretable

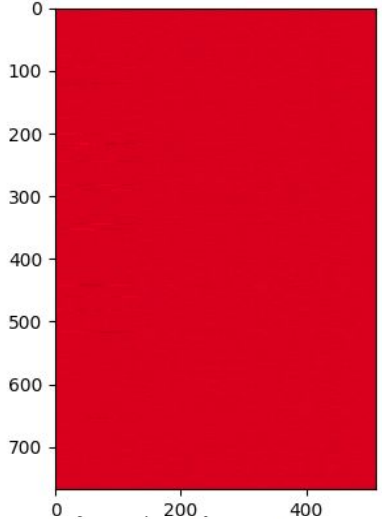
Each row in the position embedding matrix Ep represents one position embedding vector

E_p =



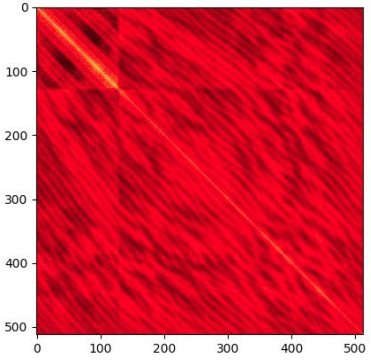
There are 512 embeddings, each with 768-dimensions in BERT base

E_p' =

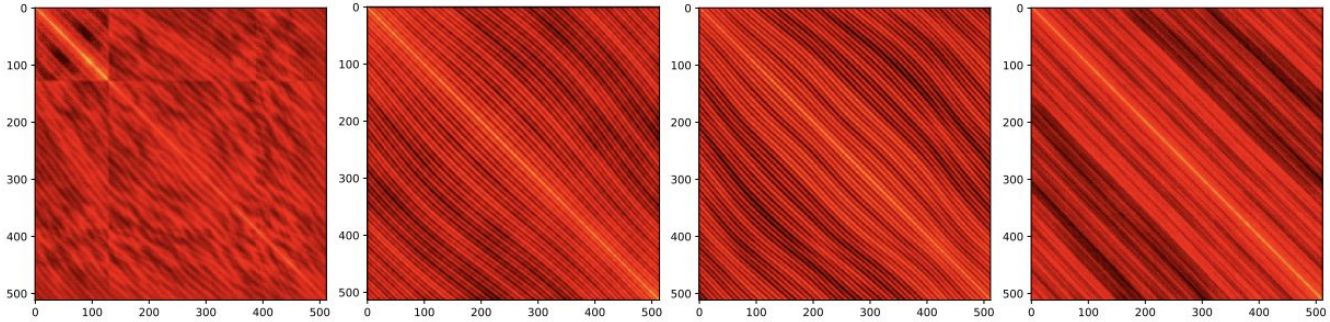


Longer sequences are disproportionately expensive because attention is quadratic to the sequence length. To speed up pretraing in our experiments, we pre-train the model with sequence length of 128 for 90% of the steps. Then, we train the rest 10% of the steps of sequence of 512 to learn the positional embeddings.

$E_p * E_p'$ =



Translation invariance in position embeddings



(a) BERT base (b) RoBERTa base (c) ALBERT base v1 (d) ALBERT xxlarge v2

Figure 1: Heatmaps visualizing the matrix $P = E_P E_P^T$ of position-embedding inner products for different models. The greater the inner product between the embeddings, the brighter the color. See appendix Figs. 4, 5 for more.

Translation invariance in position embeddings increases with training

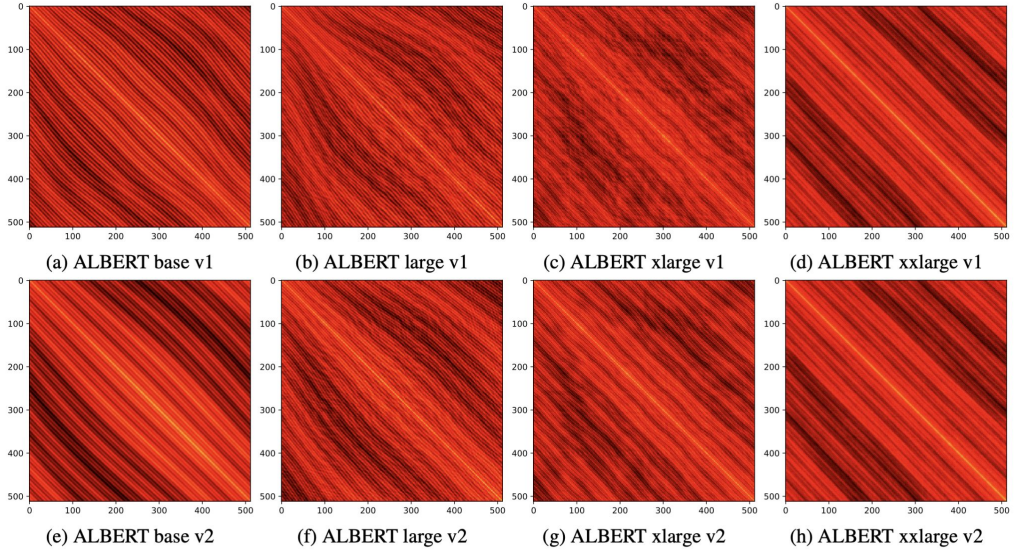


Figure 5: Visualizations of the inner-product matrix $P = E_P E_P^T \in \mathbb{R}^{n \times n}$ for different ALBERT models (Lan et al., 2020). We plot both v1 and v2 to show the progression towards increased Toeplitzness during training.

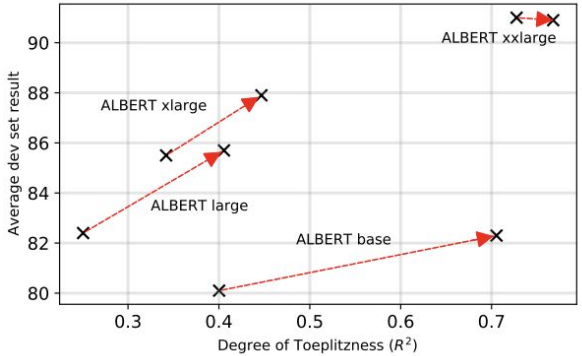


Figure 2: Scatterplot of the degree of Toeplitzness of P for different ALBERT models (v1→v2) against average performance numbers (from Lan et al.'s GitHub) over SST-2, MNLI, RACE, and SQuAD 1.1 and 2.0.

Translation invariance in positional component of self-attention

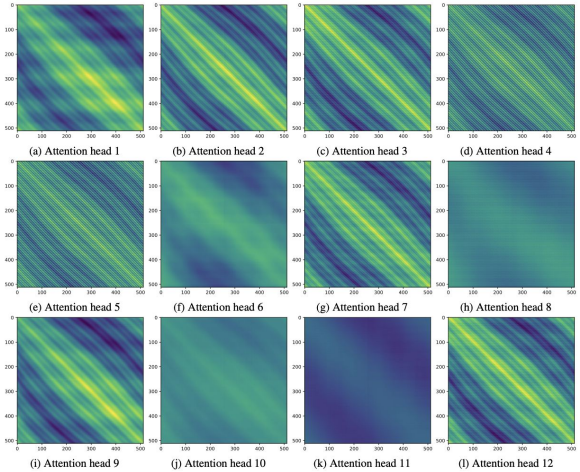


Figure 8: Values extracted from the positional attention matrices for all ALBERT base v2 first-layer attention heads. Some heads are seen to be sensitive to position, while others are not. Note that these visualizations deliberately use a different color scheme from other (red) matrices, to emphasize the fact that the matrices visualized here represent a different phenomenon and are not inner products.

TISA: Translation-Invariant Self-Attention

Modification of the self-attention equation:

$$\text{att} = \begin{cases} \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + F_P\right)V & \text{a)} \\ \text{softmax}\left(\frac{Q_W K_W^T}{\sqrt{d_k}} + F_P\right)V_W & \text{b)} \end{cases}$$

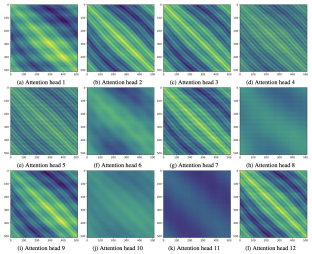


Figure 8: Values extracted from the positional attention matrices for all ALBERT base v2 first-layer attention heads. Some heads are seen to be sensitive to position, while others are not. Note that these visualizations deliberately use a different color scheme from other (red) matrices, to emphasize the fact that the matrices visualized here represent a different phenomenon and are not inner products.

TISA: Translation-Invariant Self-Attention

Modification of the self-attention equation:

$$\text{att} = \begin{cases} \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + F_P\right)V & \text{a)} \\ \text{softmax}\left(\frac{Q_W K_W^T}{\sqrt{d_k}} + F_P\right)V_W & \text{b)} \end{cases}$$

$$F_P = \begin{bmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ f_{-1} & f_0 & \cdots & f_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ f_{-(n-1)} & f_{-(n-2)} & \cdots & f_0 \end{bmatrix}$$

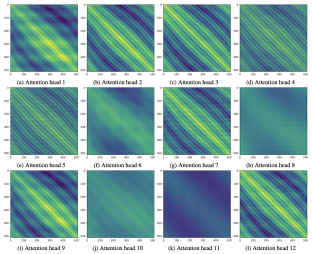


Figure 8: Values extracted from the positional attention matrices for all ALBERT base v2 first-layer attention heads. Some heads are seen to be sensitive to position, while others are not. Note that these visualizations deliberately use a different color scheme from other (red) matrices, to emphasize the fact that the matrices visualized here represent a different phenomenon and are not inner products.

TISA: Translation-Invariant Self-Attention

Modification of the self-attention equation:

$$\text{att} = \begin{cases} \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + F_P\right)V & \text{a)} \\ \text{softmax}\left(\frac{Q_W K_W^T}{\sqrt{d_k}} + F_P\right)V_W & \text{b)} \end{cases}$$

$$F_P = \begin{bmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ f_{-1} & f_0 & \cdots & f_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ f_{-(n-1)} & f_{-(n-2)} & \cdots & f_0 \end{bmatrix}$$

$$f_\theta(k) = \sum_{s=1}^{S} a_s \exp\left(-|b_s|(k-c_s)^2\right)$$

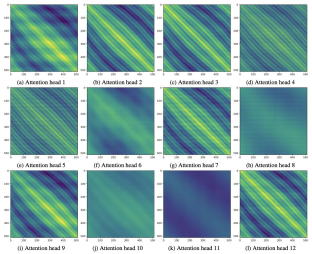


Figure 8: Values extracted from the positional attention matrices for all ALBERT base v2 first-layer attention heads. Some heads are seen to be sensitive to position, while others are not. Note that these visualizations deliberately use a different color scheme from other (red) matrices, to emphasize the fact that the matrices visualized here represent a different phenomenon and are not inner products.

Untrained* TISA improves results over ALBERT base

Task	Baseline	$S=1$	3	5	Δ	Δ%
SST-2	92.9	**93.3**	93.1	93.1	0.4	6.5%
MNLI	83.8	84.1	84.4	**84.8**	1.0	5.9%
QQP	88.2	88.0	**88.3**	**88.3**	0.1	1.2%
STS-B	90.3	**90.4**	90.0	**90.4**	0.1	1.5%
CoLA	57.2	57.0	56.5	**58.5**	1.3	2.9%
MRPC	89.6	**90.1**	89.0	**90.1**	0.5	5.3%
QNLI	91.6	**91.7**	91.4	91.6	0.1	0.4%
RTE	72.9	71.1	**73.6**	**73.6**	0.7	2.7%

(a) ALBERT base v2 models with position embeddings

Task	Baseline	$S=1$	3	5	Δ	Δ%
SST-2	85.1	85.9	85.8	**86.0**	0.9	6.2%
MNLI	78.8	80.9	81.4	**81.6**	2.8	13.4%
QQP	86.3	86.2	86.5	**86.8**	0.5	3.4%
STS-B	89.0	89.0	**89.1**	**89.1**	0.1	0.3%
MRPC	82.8	83.1	**83.3**	83.1	0.5	3.3%
QNLI	86.6	87.2	87.4	**87.7**	1.1	7.8%
RTE	62.1	61.7	62.5	**62.8**	0.7	1.9%

(b) ALBERT base v2 models without position embeddings

Table 3: GLUE task dev-set performance (median over 5 runs) with TISA (S kernels) and without (baseline). Δ is the maximum performance increase in a row and Δ% is the corresponding relative error reduction rate.

$$\text{att} = \begin{cases} \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + F_P\right)V & \text{a)} \\ \text{softmax}\left(\frac{Q_W K_W^T}{\sqrt{d_k}} + F_P\right)V_W & \text{b)} \end{cases}$$

*: We perform a custom initialization procedure, but conducted no language modeling pre-training beyond the pre-trained baseline we started from

Thank you for listening

Paper available at: https://arxiv.org/abs/2106.01950
Code available at: https://github.com/ulmewennberg/tisa