FORMATTING INSTRUCTIONS FOR ICLR 2020 CONFERENCE SUBMISSIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

The abstract paragraph should be indented 1/2 inch (3 picas) on both left and right-hand margins. Use 10 point type, with a vertical spacing of 11 points. The word ABSTRACT must be centered, in small caps, and in point size 12. Two line spaces precede the abstract. The abstract must be limited to one paragraph. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

1 INTRODUCTION

Recent advances in Natural Language Processing (NLP) are due to pretraining of large models based on the Transformers architecture (Vaswani et al., 2017). How Transformers process input text differs significantly from Recurrent Neural Networks as it does not ingest the text from left to right (or right to left) but each token representation can leverage all other tokens of the sentence. The core of Transformers resides in the Self-Attention mechanism. The attention mechanism was originally introduced in Neural Machine Translation to better handle long range dependencies and align pairs of sentences to translate (Bahdanau et al., 2015).

The success of such architectures in NLP made us reconsider the absolute dominance of the Convolutional Layers in vision and question its supremacy. Applying the transformer architecture from text to images is only stepping up one dimension but involves some technical challenges of scale. In section 2 we review these challenges and present advances using Transformers on images.

We show in section 3 that this success is not so surprising. In fact, Self-Attention layers have the expressive power to encode CNN layers under very basic conditions: (i) positional encoding must be relative to the position of the computed value, (ii) self-attention requires multiple heads (iii) positional encoding can be fixed to pixel position difference with second order combinations. We show that Self-Attention layers generalize CNN layers to not only learn the filters but also the position of the "attended" pixels.

JB: Need to introduce multi-head before

JB: I don't show that (iii) is necessary but it's an example, not sure how to state that it does not require a crazy encoding schema.

The ability for Self-Attention layer to express CNN does not ensure that such filters are learnable. Through experiments, we demonstrate that standalone self-attention models can perform as well as well established ResNet. Our experiments (in section 4) focus more on the interpretation of this expressive power and investigate different type of positional encoding. It is particularly interesting to study such positional encoding schema on images, where we have a good intuition, to then transfer the lessons learned to NLP, where distances between words and absolute positions in the sentence can vary greatly between languages and seem less concrete.

2 BACKGROUND ON ATTENTION MECHANISM AND RELATED WORK

We first recall the formulation of standard CNN layers and Self-Attention layers using unified notation. We then review how recent lines of work have incorporated Self-Attention layers into classical ResNet architecture for vision (Ramachandran et al., 2019; Bello et al., 2019).

2.1 Convolution Layer

A convolution layer is defined by a kernel size K (assuming square kernel for simplicity), a number of input channel C_{in} and a number of output channel C_{out}. It is parametrized by a kernel tensor $\mathbf{W} \in \mathbb{R}^{K \times K \times C_{out} \times C_{in}}$ and a bias vector $\boldsymbol{b} \in \mathbb{R}^{C_{out}}$. Given an input image $\mathbf{X} \in \mathbb{R}^{W \times H \times C}$ of width W, height H and C channels, the output of the convolution layer is given by,

$$\mathbf{Y}_{i,j,:} = \boldsymbol{b} + \sum_{(m,n) \in \{\lfloor (K-1)/2 \rfloor, \dots, \lfloor K/2 \rfloor\}^2} \boldsymbol{W}_{m,n,:,:} \mathbf{X}_{i+m,j+n,:} \tag{1}$$

2.2 Self-Attention Layer

A Self-Attention layer is defined by its hidden dimension d_h and its number of heads K. It is parametrized by $\mathbf{W}^Q \in \mathbb{R}^{K \times d \times d_h}$ query layer, $\mathbf{W}^V \in \mathbb{R}^{K \times d \times d_h}$ key layer, $\mathbf{W}^V \in \mathbb{R}^{K \times d \times d_h}$ value layer, a projection layer $\mathbf{W}^Q \in \mathbb{R}^{K \times d_h \times d}$ and output layer $\boldsymbol{W}^O \in \mathbb{R}^{(K d_h) \times d}$. Applied to an input $\boldsymbol{X} \in \mathbb{R}^{T \times d}$, the output of the Self-Attention layer is computed as,

JB: same as size of the CNN kernel, we point that they are related (K^2) but need clarify notation

$$\boldsymbol{A}_k = (\boldsymbol{X}\boldsymbol{W}_k^Q)(\boldsymbol{X}\boldsymbol{W}_k^K)^\top , \tag{2}$$

$$\boldsymbol{H}_{t,:} = \sum_{k \in \{1,\dots,K\}} \operatorname{softmax}(\boldsymbol{A}_k)\, \boldsymbol{X}\boldsymbol{W}_k^V \boldsymbol{W}_k^P , \tag{3}$$

$$\boldsymbol{Y}_{t,:} = \boldsymbol{H}_{t,:}\boldsymbol{W}^O + \boldsymbol{X}_{t,:} , \tag{4}$$

where the $\operatorname{softmax}(\cdot)$ is taken over all the dimensions of the tensor but the first one. For simplicity, we exclude the batch normalization layers and constant factors.

JB: check dimensions and readability

2.3 Attention Mechanism in Vision

A recent trend emerged using Self-Attention to solve vision tasks. It is mainly motivated by the ability of attention mechanism to model long range dependencies across different part of images and help solving Visual Question Answering tasks. Successful works have reached the level of accuracy of ResNets on classification tasks for Cifar and ImageNet datasets. Along with these impressive results, Bello et al. (2019) advocate for using Self-Attention layers along with classical convolutions to reach best performance. Ramachandran et al. (2019) show that even standalone Self-Attention can be enough and propose strategies to downsample the image to reduce the number of pixel-to-pixel attention score to compute and store in memory.

JB: cite

There are two main challenges in reusing Self-Attention, originally designed for text, for images:

Positional Encoding.

Model	type of positional encoding			relative
	sinusoids	learned	quadratic	
Vaswani et al. (2017)	✓			
Radford et al. (2018)		✓		
Devlin et al. (2018)		✓		
Dai et al. (2019)	✓			✓
Yang et al. (2019)	✓			✓
Bello et al. (2019)		✓		✓
Ramachandran et al. (2019)		✓		✓
Our work			✓	✓

Table 1: Types of positional encoding used by transformers models applied to text (*top*) and images (*bottom*). When multiple encoding types have been tried, we report the one advised by the authors.

Downsampling.

3 Self-Attention Layer Can Implement Convolution Layer

Our goal is to show that attention layers have the expressive power to learn convolution filter and that we can learn such filters in practice. The ability of the Attention mechanism to encode spacial filters (3×3 kernels) depends heavily on how we encode position in the image.

3.1 Relative Position Encoding and Translation Equivariance

Note that the Self-Attention mechanism (4) is equivarient to reordering. It means that shuffling the order of the tokens does not affect the meaning extracted by the model. To alleviate this problem, an embedding is learned for each position in a sentence (or pixel in an image) and added to the representation of the token before applying self-attention.

$$\boldsymbol{X}_{t,:} = \boldsymbol{E}_{x_t} + \boldsymbol{P}_t \,, \tag{5}$$

where $\boldsymbol{E}_{x_t}$ is the learned vector representation of the x_t token and $\boldsymbol{P}$ is a position embedding matrix defined up to a maximum length, but it could be any function that returns a vector representation of the position.

Relative position encoding was introduced by Dai et al. (2019) in TransformerXL. The main idea is to use the position difference between the query token (token we compute the representation of) and the key token (token we attend) instead of only the absolute position of the key token. The computation of the attention coefficients (eq. (2)) can be decomposed as follows:

$$\mathbf{A}^{\text{abs}}_{i,j} = (\boldsymbol{E}_{x_i} + \boldsymbol{P}_i)\boldsymbol{W}^Q(\boldsymbol{W}^K)^\top(\boldsymbol{E}_{x_j} + \boldsymbol{P}_j)^\top \tag{6}$$

$$= \underbrace{\boldsymbol{E}_{x_i}\boldsymbol{W}^Q(\boldsymbol{W}^K)^\top\boldsymbol{E}_{x_j}^\top}_{(a)} + \underbrace{\boldsymbol{E}_{x_i}\boldsymbol{W}^Q(\boldsymbol{W}^K)^\top\boldsymbol{P}_j^\top}_{(b)} + \underbrace{\boldsymbol{P}_i\boldsymbol{W}^Q(\boldsymbol{W}^K)^\top\boldsymbol{E}_{x_j}}_{(c)} + \underbrace{\boldsymbol{P}_i\boldsymbol{W}^Q(\boldsymbol{W}^K)^\top\boldsymbol{P}_j}_{(d)} \,. \tag{7}$$

To simplify the notation, we considered only one attention head and dropped the subscript k. They replace all absolute position encoding with relative ones:

JB: maybe better notation $\boldsymbol{K}$ for $\boldsymbol{W}^K$

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\boldsymbol{E}_{x_i}^\top\boldsymbol{W}_q^\top\boldsymbol{W}_{k,E}\boldsymbol{E}_{x_j}}_{(a)} + \underbrace{\boldsymbol{E}_{x_i}^\top\boldsymbol{W}_q^\top\boldsymbol{W}_{k,R}\mathbf{R}_{i-j}}_{(b)} + \underbrace{\boldsymbol{u}^\top\boldsymbol{W}_{k,E}\boldsymbol{E}_{x_j}}_{(c)} + \underbrace{\boldsymbol{v}^\top\boldsymbol{W}_{k,R}\mathbf{R}_{i-j}}_{(d)} \tag{8}$$

Simply show the translation equivariance with relative encoding.

3.2 Attention on a Pixel at any Relative Position

$$\mathsf{H}_{i,j,:} = \sum_{k\in\{1,\dots,K\}} \delta\mathsf{X}\boldsymbol{W}_k^V \tag{9}$$

define dirac $\delta_{i,j}$ is a matrix whose elements are zero everywe

To mimic the CNN computation, each attention head focuses on a pixel at a given relative position. Hence reproducing a 3×3 kernel requires 9 attention heads and the ability for each of them to perfectly attend on one pixel at a relative position. Given that spatial convolution filters are not conditioned on the input data, we set $\boldsymbol{W}_{k,E}$ and $\boldsymbol{W}_{k,R}$ to 0, leaving only the (d) term in equation 8.

$$\mathbf{A}^{\text{conv}}_{i,j} = \boldsymbol{v}^\top\boldsymbol{W}_{k,R}\mathbf{R}_{i-j} \tag{10}$$

As both $\boldsymbol{v}$ and $\boldsymbol{W}_{k,R}$ are learnable parameters, we simplify the expression to learn a single vector $\mathbf{t}_h$ for each head h.

How to discern relative position? The goal is to reproduce the 3×3 pattern of CNN kernel with attention instead. Each head should be able to chose which pixel to attend based on its relative position to the query token x_i. The most basic attention mechanism is the dot product $(\boldsymbol{p}_i - \boldsymbol{p}_j)^\top\mathbf{t}_h$ between the relative positions of the input pixels and the relative target positions $\mathbf{t}_h = (x_h, y_h)^\top$. The dot product is large for pixels in the same direction as $\mathbf{t}_h$, however it also grows for farther pixels. To be able to focus on a relative pixel position (direction and distance), we propose the following relative position encoding

$$\boldsymbol{R}_\Delta = \begin{pmatrix}\Delta_x & \Delta_y & \Delta_x^2 & \Delta_y^2 & 1 & 1\end{pmatrix}^\top \tag{11}$$

Instead of learning the target vector $\mathbf{t}_h$, we parametrize it for each head to attend to the relative pixel position (x_h, y_h),

$$\mathbf{t}_h = -\alpha_h \begin{pmatrix}-2x_h & -2y_h & 1 & 1 & x_h^2 & y_h^2\end{pmatrix}^\top \tag{12}$$

The attention coefficient for head h between pixel i and j is given by the dot product,

$$\boldsymbol{R}_\Delta^\top \mathbf{t}_h = -\alpha_h((\Delta_x - x_h)^2 + (\Delta_y - y_h)^2), \tag{13}$$

with $\Delta = \boldsymbol{p}_i - \boldsymbol{p}_j$. The maximum attention coefficient is 0 when $\Delta = (x_h, y_h)$, i.e. the center of attention of a given head. The α coefficient controls how spiky the attention is (analogous to temperature in the softmax). Figure 1 gives a representation of the attention weights for a fixed query pixel i and different α and (x_h, y_h).

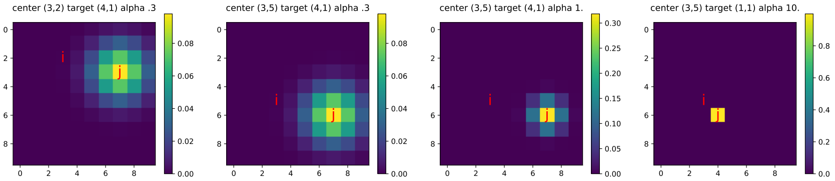


Figure 1: Attention coefficient for different center i and target j with varying α parameters. Target designate the relative position that the head attends to.

Discussion. Some relations can be drawn to RBF kernels centered at the target pixels. We are expressing a more general form of convolutions where *(i)* the input pixels do not follow a grid shape but has any pattern and *(ii)* the inputs are not single pixels but weighted averages of local patches in the image.

Simplification. One further simplification is to remove the constant terms of $\boldsymbol{R}_\Delta$ because the softmax (renormalization) is shift invariant and constant terms for a head are useless.

3.3 Non Isotropic Gaussian

In this section, we present an extension of the relative positional encoding introduced above to model non-isotropic Gaussians.

$$f_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\boldsymbol{x}) = \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right) \tag{14}$$

We drop the normalization factor of the Gaussian distribution as it is replaced by the softmax of the attention mechanism. The matrix $\boldsymbol{\Sigma}^{-1}$ must be positive semi-definite. We parametrize it as

$$\boldsymbol{\Sigma}^{-1} = (\boldsymbol{\Sigma}^{-1/2})^\top \boldsymbol{\Sigma}^{-1/2} = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \tag{15}$$

Our goal is to write the exponent as a dot product of two vectors:

- one vector $R_{\boldsymbol{x}}$ dependent only on $\boldsymbol{x}$, the relative positional encoding,
- one vector $\mathbf{t}_h$ dependent only on $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, the attention head.

Model	Number of parameters
ResNet10	4.9M
BERT 4 layers	1.85M
BERT 6 layers	9.56M

Table 2: Number of parameters per model.

$$R_{\boldsymbol{x}} = \begin{pmatrix} x_1 & x_2 & x_1 & x_2 & x_1^2 & x_2^2 & x_1 x_2 & 1 & 1 & 1 \end{pmatrix} \tag{16}$$

$$\mathbf{t}_h = -\frac{1}{2}\begin{pmatrix} -2a\mu_1 & -2c\mu_2 & -2b\mu_2 & -2b\mu_1 & a & c & 2b & a\mu_1^2 & c\mu_2^2 & 2b\mu_1\mu_2 \end{pmatrix} \tag{17}$$

$$R_{\boldsymbol{x}}^\top \mathbf{t}_h = -\frac{1}{2}\left[a(x_1-\mu_1)^2 + 2b(x_1-\mu_1)(x_2-\mu_2) + c(x_2-\mu_2)^2\right] = -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \tag{18}$$

We can check that setting $b = 0$ and $a = c = \alpha$ recovers the expression of the isotropic Gaussian derived above.

3.4 Implications to Sinusoid and Learned Positional Encoding

3.5 Exploit Attention on Features?

- Does the terms (a) (b) (c) allow to condition the CNN filters on the input data?

4 Experiments

We want to validate that not only Transformer architecture can express CNN filters but that it can also learn such filters with SGD from the data. It has already been shown by (Bello et al., 2019) that adding attention features to CNNs can improve performence on Cifar-100 and ImageNet. We want to stick to a fully attentional model, and show that it matches its siblings ResNet of equivalent depth.

4.1 Performance Against ResNet

We design a smaller ResNet (i.e. ResNet10) which have similar number of layers as our transformer. The sizes of the models are displayed in Table 2.

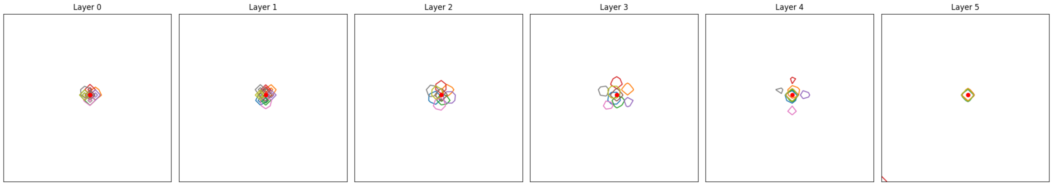


Figure 2: Contours of attention weights per layer for the 6 layers fully attentional model.

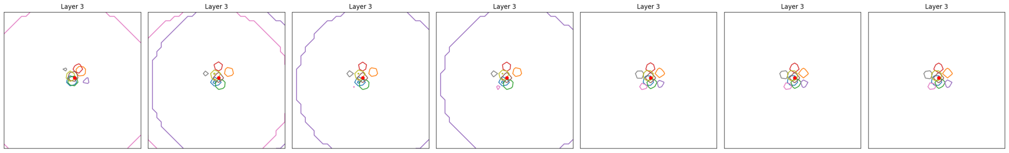
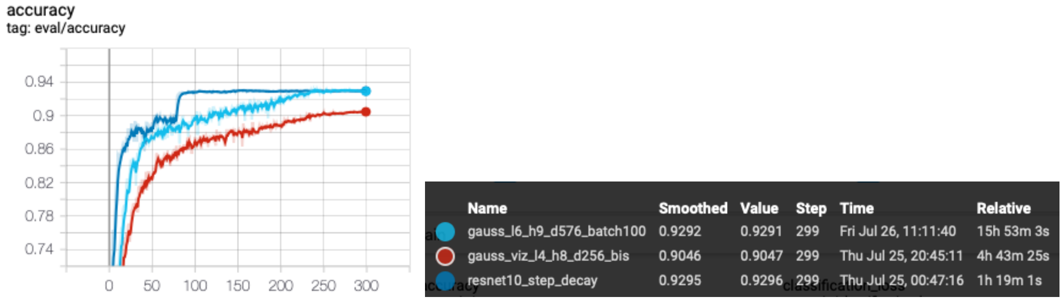


Figure 3: Contours of attention weights during training (300 epochs) at layer 3.



	Name	Smoothed	Value	Step	Time	Relative
●	gauss_l6_h9_d576_batch100	0.9292	0.9291	299	Fri Jul 26, 11:11:40	15h 53m 3s
●	gauss_viz_l4_h8_d256_bis	0.9046	0.9047	299	Thu Jul 25, 20:45:11	4h 43m 25s
●	resnet10_step_decay	0.9295	0.9296	299	Thu Jul 25, 00:47:16	1h 19m 1s

Figure 4: Evaluation accuracy on CIFAR-10 of a small ResNet and two Gaussian Attention models.

5 Discussion

Acknowledgments

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V. Le. Attention Augmented Convolutional Networks. *arXiv:1904.09925 [cs]*, April 2019.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *CoRR*, abs/1901.02860, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2018. URL `https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf`.

Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *CoRR*, abs/1906.05909, 2019. URL `http://arxiv.org/abs/1906.05909`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL `http://arxiv.org/abs/1906.08237`.