Factorized Attention: Self-Attention with Linear Complexities

Shen Zhuoran* shenzhuoran
Zhang Minyuan* zhangmingyuan
Yi Shuai yishuai
Yan Junjie yanjunjie
Zhao Haiyu zhaohaiyu

SenseTime Research
@sensetime.com

Abstract

Recent works have been applying self-attention to various fields in computer vision and natural language processing. However, the memory and computational demands of existing self-attention operations grow quadratically with the spatiotemporal size of the input. This prohibits the application of self-attention on large inputs, e.g.*, long sequences, high-definition images, or large videos. To remedy this, this paper proposes a novel factorized attention (FA) module, which achieves the same expressive power as previous approaches with substantially less memory and computational consumption. The resource-efficiency allows more widespread and flexible application of it. Empirical evaluations on object recognition demonstrate the effectiveness of these advantages. FA-augmented models achieved state-of-the-art performance for object detection and instance segmentation on MS-COCO. Further, the resource-efficiency of FA democratizes self-attention to fields where the prohibitively high costs currently prevent its application. The state-of-the-art result for stereo depth estimation on the Scene Flow dataset exemplifies this.*

1. Introduction

Modeling long-range dependencies has been a central difficulty for deep learning. The most successful architectures, *e.g.*, convolutional and recurrent neural networks, process a local neighborhood at a time. To model long-range dependencies, these operations must be applied repeatedly to increase the receptive field. To mitigate this, [1] introduced the self-attention mechanism, which computes the response at every location as a weighted average of features at all locations in the previous layer. Since its advent, the method has been remarkably successful. Self-attention-based models now holds state-of-the-art records on virtually all tasks in natural language processing (NLP) [31, 23, 9]. The non-local module [33], an adaptation of self-attention for vision networks, achieved state-of-the-art performance on video classification [33] and generative adversarial image modeling [36, 3] and demonstrated significant improvements on object detection [33], instance segmentation [33], person re-identification [17], and image deraining [15], among other tasks.

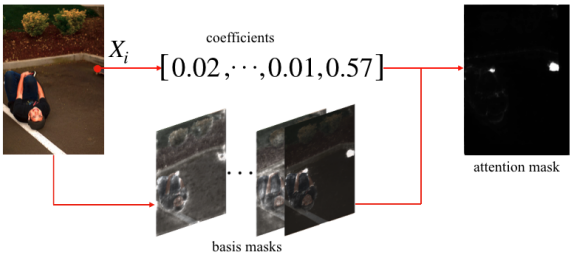

Figure 1. **An illustration of the conceptual mechanism of factorized attention.** The red dot on the left image identifies the point of interest. The basis masks are shared by all pixels, while the coefficients belong to X_i. Conceptually, aggregating the basis masks with the coefficients gives the attention mask for X_i, but the mask will not be computed in practice for memory and computational complexity optimization.

The mechanism's success relies on its ability to efficiently model long-range dependencies. Without self-attention, a 256x256 input image requires 128 consecutive 3x3 convolutions to reach a global receptive field and ensure the ability to model dependencies between any two pixels. Stacking many layers leads to computational inefficiency. Besides this, relying on a deep network to propagate information between two distant points might cause optimization difficulties. Further, multi-hop dependency modeling multiplies the resource requirements. For example, modeling 3-hop dependencies on a 256x256 image demands 384 3x3 convolutions. This further reinforces the computational inefficiency and optimization difficulties. In contrast, a single self-attention module expands the receptive field to the entire input by connecting every location with all locations in the previous layer. Using self-attention to efficiently model long-range dependencies allows convolutions and recurrent

*Equal contribution

operations to focus on local dependency modeling in which they specialize.

However, global dependency modeling on large inputs, *e.g*. long sequences, high-definition images, and large videos, is still an open problem. The quadratic[1] memory and computational complexities of existing self-attention modules inhibit their application on such inputs. For instance, a non-local module uses over 1 GB of memory and over 50 GMACC[2] of computation for a 128x128 feature map or over 68 GB and over 3.2 TMACC for a 64x64x32 video feature volume. This constrains the application of self-attention to the low-resolution parts of models [33, 36, 3] and prohibits its use for resolution-sensitive and resource-hungry tasks.

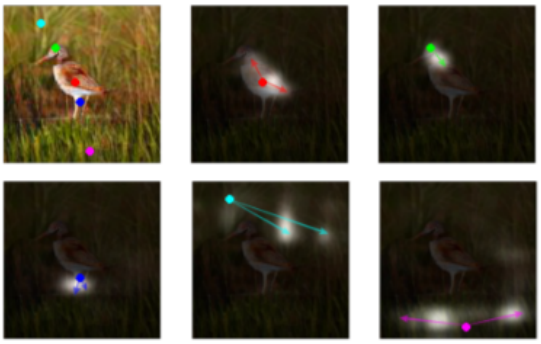

Figure 2. **An illustration of the learned attention masks in a non-local module.** The first image identifies five query locations with colored dots. Each of the subsequent images illustrates the attention map for one of the locations. Adapted from [36].

The above two points, namely the complementarity between large input sizes and self-attention and the quadratic memory and computational complexities of existing self-attention modules, greatly motivate the exploration for a resource-efficient self-attention algorithm. An investigation into the non-local module revealed an intriguing phenomenon. The attention masks at each location, despite generated independently, are not entirely unrelated. As [33] and [36] analyzed, the attention mask of a location mainly focuses on regions with semantic relations to it. Figure 2 shows the learned attention masks in a non-local module. When generating the image of a bird before a bush, pixels on the bird's legs tend to attend to other leg pixels for structural consistency. Similarly, body pixels mainly attend to the body, and background pixels focus on the bush. In this example, a body mask, a leg mask, and several background masks might be present in an ideal set of basis masks.

[1]The complexities are quadratic with respect to the spatiotemporal size of the input, or quartically w.r.t. the side length of an input image, or sextically w.r.t. the dimension of an input video.

[2]MACC stands for multiply-accumulation. 1 MACC means 1 multiplication or addition operation.

This observation inspired the design of the factorized attention module. Given an input, the module generates a set of basis masks, a coefficient for each mask at every location, and a value feature map. Figure 1 shows the conceptual structure of the module. Scaling each mask by its coefficient at a location and summing the scaled masks gives the attention mask for the location. In practice, the module first aggregates the value map by each basis mask and then combines the aggregated contextual features at each location by the coefficients. This procedure avoids the generation of the huge pairwise attention matrix containing the attention mask for every location and hence removes the quadratic terms in the module's memory and computational complexities.

The principal contribution of this paper is the factorized attention module, which:

1. has linear memory and computational complexities in terms of the spatiotemporal size of the input, as will be shown in section 3.4;

2. possesses the same expressive power as the previous modules, as will be proven in section 3.3;

3. provides substantial performance boosts on resolution-sensitive tasks, as demonstrated by the state-of-the-art results for object detection and instance segmentation on MS-COCO 2017 in section 4.1; and

4. democratizes the self-attention mechanism to numerous previously inapplicable fields, as exemplified by the state-of-the-art result for stereo depth estimation on the Scene Flow dataset in section 4.2.

2. Related Works

2.1. The Self-Attention Mechanism

[1] proposed the initial formulation of the self-attention mechanism to improve word alignment in machine translation. Successively, [31] proposed to completely replace the recurrent operation with self-attention and named the resultant architecture the Transformer. They achieved state-of-the-art performance on machine translation and constituency parsing with the architecture. Following this work, [23] pretrained a deep Transformer on a large corpus for language modeling and achieved state-of-the-art performance on a broad set of NLP tasks, including natural language inference, semantic similarity, and reading comprehension. Subsequently, [9] proposed to use masked language modeling and next sentence prediction for pretraining to incorporate bidirectional information. With this work, Transformers now hold the state-of-the-art records on virtually all NLP tasks and outperform humans on many.

[33] first adapted the self-attention mechanism for computer vision. They achieved state-of-the-art performance on

video classification and demonstrated significant improvements on video classification, object detection, instance segmentation, and pose estimation. The paper named the proposed network component the non-local module. Subsequent works applied it to various fields in computer vision, including image restoration [20], de-raining [15], video person re-identification [17], salient segmentation [14], and cardiac motion scoring [35]. Notably, the SAGAN [36] and BigGAN [3] incorporated the non-local module to generative adversarial networks and achieved substantial advancements of the state-of-the-art.

2.2. Other Types of Visual Attention

Besides self-attention, there is separate set of techniques referred to as attention. This section refers to them as scaling attention. While self-attention is effective for global dependency modeling, scaling attention focuses on emphasizing important features and suppressing uninformative ones. [32] proposed to replace the ResBlock with a complex attention module as the backbone component of convolutional networks. The module consists of a convolutional branch and an hourglass attention branch. It multiplies the attention map generated by the hourglass branch to the output of the convolutional branch to achieve effective scaling. This architecture outperformed the ResNet while consuming less resources. Compared to this method, the squeeze-and-excitation (SE) module [13] is a significantly more economical approach. An SE block uses global average pooling and a linear layer to compute a scaling factor for each channel. The low cost allows the insertion of an SE block at every layer of a network, which allowed SE-enhanced models to achieve state-of-the-art performance on image classification and substantial improvements on scene segmentation and object detection. The CBAM [34] built upon the SE module by adding global max pooling beside global average pooling and introducing an extra spatial attention module. The CBAM further improved SE's performance on image classification and object detection.

2.3. Object Detection

Since the introduction of the R-CNN [11], deep learning methods have achieved remarkable progress on object detection. Most CNN detectors belong to one of the two categories: two-stage detectors and single-stage detectors. Two-stage models first generate proposals that likely contain objects and then refine the box coordinates and the classification. Notable two-stage methods after the R-CNN include the Fast R-CNN [10], the Faster R-CNN [27], and the R-FCN [7]. In contrast, single-stage detectors directly output detection boxes from the features extracted by the backbone network. The SSD [21] and the YOLO family [24, 25, 26] are popular architectures in this category.

Various works have attempted to improve the detection pipeline. [18] proposed the FPN which aggregates features across scales and network depths. The Mask R-CNN [12] incorporated a segmentation head to perform instance segmentation together with detection. [8] introduced the deformable convolution that replaces the fixed square receptive fields of the standard convolution by learnable receptive fields. Cascade R-CNN [4] stacks multiple detectors with increasing IoU thresholds to adapt to different IoU requirements at test time. RetinaNet [19] introduced the focal loss to address class imbalance and tackle hard examples.

3. Method

3.1. A Generic Formulation of Conventional Self-Attention

Before presenting factorized attention, this section will first give a generic formulation of the self-attention mechanism, to facilitate straightforward comparisons between the factorized attention module and previous approaches.

Unlike the convolutional or recurrent operation, which processes a local region at a time, the self-attention operation treats all positions the same. Therefore, any 3D feature map of size $h \times w \times c$ or 4D feature volume of size $h \times w \times d \times c$ can be regarded as a matrix of size $hw \times c$ or $hwd \times c$ in the process. For simplicity, the subsequent analysis will assume all input, output, and intermediate representations are matrices.

Following is a formulation of the self-attention mechanism. For an input matrix $\boldsymbol{X}$ of size $s \times c$, it is first passed through three separate functions $\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V}$, which generate the key $\boldsymbol{K}(\boldsymbol{X})$, the query $\boldsymbol{Q}(\boldsymbol{X})$, and the value $\boldsymbol{V}(\boldsymbol{X})$, respectively. Then, an affinity function $\boldsymbol{A}$ takes in $\boldsymbol{K}(\boldsymbol{X})$ and $\boldsymbol{Q}(\boldsymbol{X})$ and produces an attention map $\boldsymbol{A}(\boldsymbol{K}(\boldsymbol{X}), \boldsymbol{Q}(\boldsymbol{X}))$ of size $s \times s$. Finally, the attention map is multiplied to the value $\boldsymbol{V}(\boldsymbol{X})$ to obtain the attended output. In summary, the self-attention module can be expressed as

$$\boldsymbol{S}(\boldsymbol{X}) = \boldsymbol{A}(\boldsymbol{K}(\boldsymbol{X}), \boldsymbol{Q}(\boldsymbol{X}))\, \boldsymbol{V}(\boldsymbol{X}). \tag{1}$$

A self-attention module can be optionally augmented with an additional linear reprojection of the output to change the number of output channels and/or a residual connection to ease the gradient flow. These modifications are trivial, and the formal formulation omits them for brevity.

When $\boldsymbol{X}$ is a 3D feature map or 4D feature volume, $\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V}$ are linear layers, and $\boldsymbol{A}$ is realized as matrix multiplication, equation (1) captures the non-local block [33] with the necessary reshaping, projection, and residual shortcuts added. When $\boldsymbol{X}$ is an input feature sequence augmented with sinusoidal positional embeddings, $\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V}$ are linear layers, and $\boldsymbol{A}$ is matrix multiplication, (1) captures the Transformer [31] architecture.

3.2. Factorized Attention

The factorized attention module also passes the input $\boldsymbol{X}$ through three separate functions, $\boldsymbol{C}, \boldsymbol{B}, \boldsymbol{V}$. The three generated matrices are referred to as the coefficients $\boldsymbol{C}(\boldsymbol{X})$, the basis $\boldsymbol{B}(\boldsymbol{X})$, and the value $\boldsymbol{V}(\boldsymbol{X})$, respectively. Then, a context extraction function $\boldsymbol{N}$ takes in $\boldsymbol{B}(\boldsymbol{X})$ and $\boldsymbol{V}(\boldsymbol{X})$ and produces the attention context matrix $\boldsymbol{N}(\boldsymbol{B}(\boldsymbol{X}), \boldsymbol{V}(\boldsymbol{X}))$. Finally, the coefficients are multiplied to the context to give the attended output. The entire process is summarized as

$$\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{C}(\boldsymbol{X})\, \boldsymbol{N}(\boldsymbol{B}(\boldsymbol{X}), \boldsymbol{V}(\boldsymbol{X})). \tag{2}$$

Similar to conventional self-attention modules, a factorized attention block supports optional reprojection of the output and residual connections.

In equation (2), the functions $\boldsymbol{C}, \boldsymbol{B}, \boldsymbol{V}, \boldsymbol{N}$ are generic. The following subsections will introduce several concrete instantiations of the factorized attention module. To simplify the presentation, the subsections will assumed $\boldsymbol{X}$ is of size $s \times c$.

3.2.1 Factorized Scaled Dot-Product Attention

The scaled dot-product version of factorized attention implements $\boldsymbol{C}$, $\boldsymbol{B}$, and $\boldsymbol{V}$ as linear layers (1x1 convolutions for 3D inputs or 1x1x1 for 4D). The outputs from $\boldsymbol{C}$ and $\boldsymbol{B}$ must agree in the number of channels. The implementation of $\boldsymbol{N}$ multiplies $\boldsymbol{B}(\boldsymbol{X})$ with $\boldsymbol{V}(\boldsymbol{X})$ and divides the product by the number of channels of $\boldsymbol{B}(\boldsymbol{X})$ for normalization. The module is formally defined as

$$\boldsymbol{F}_{\text{dot}}(\boldsymbol{X}) = \frac{\boldsymbol{X}\boldsymbol{C}\left((\boldsymbol{X}\boldsymbol{B})^{\mathrm{T}}\boldsymbol{X}\boldsymbol{V}\right)}{b}, \tag{3}$$

for b an integer hyperparameter, $\boldsymbol{C}, \boldsymbol{B}$ $c \times b$ matrices, and $\boldsymbol{V}$ a $c \times c$ matrix.

3.2.2 Factorized Gaussian Attention

In the factorized Gaussian attention module, $\boldsymbol{C}$ and $\boldsymbol{B}$ each instantiates as a linear layer followed by a softmax normalization. Their outputs must have equal numbers of channels. $\boldsymbol{C}$ performs the softmax operation across all channels at each location, while $\boldsymbol{B}$ performs softmax across all locations in each channel. This ensures that the attention map $\boldsymbol{C}(\boldsymbol{X})\,\boldsymbol{B}(\boldsymbol{X})^{\mathrm{T}}$ (which will not be actually computed) satisfies the property that each row (corresponding to a location) sums up to 1, as the attention map in a conventional Gaussian self-attention module satisfies. The implementations of $\boldsymbol{V}$ and $\boldsymbol{N}$ are identical to those in the scaled dot-product instantiation. The definition of this version is

$$\boldsymbol{F}_{\text{gauss}}(\boldsymbol{X}) = \sigma_{\text{c}}(\boldsymbol{X}\boldsymbol{C})\left(\sigma_{\text{s}}\left((\boldsymbol{X}\boldsymbol{B})^{\mathrm{T}}\right)\boldsymbol{X}\boldsymbol{V}\right), \tag{4}$$

for b an integer hyperparameter, $\boldsymbol{C}, \boldsymbol{B}$ $c \times b$ matrices, $\boldsymbol{V}$ a $c \times c$ matrix, and $\sigma_{\text{c}}, \sigma_{\text{s}}$ softmax normalization along the channel and location dimensions, respectively.

3.2.3 Factorized Multi-Head Attention

The multi-head attention is a method to augment an attention module, proposed by [31]. A multi-head FA module accepts a hyperparameter h as the number of heads. The channel depth of the outputs of $\boldsymbol{C}$, $\boldsymbol{B}$, and $\boldsymbol{V}$ all reduce to $\frac{1}{h}$ of the original. Then, h such attention modules are applied on the input in parallel. Their results are then concatenated along the channel dimension, making the output channel count the same as in a single-head attention module. The multi-head module is formally expressed as

$$\boldsymbol{F}_{\text{multi}}(\boldsymbol{X}) = \bigodot_{i=0}^{h-1} \boldsymbol{F}_i(\boldsymbol{X}), \tag{5}$$

for $\{\boldsymbol{F}_i\}_{i=0}^{h-1}$ a set of independent single-head attention modules and $\bigodot$ the concatenation operator.

3.3. Equivalence between Factorized and Conventional Self-Attention

The first subsection will present a proof that the factorized and conventional variants of scaled dot-product attention are mathematically equivalent. The next subsection will conduct analysis on other variants.

3.3.1 Proof for Scaled Dot-Product Attention

Assume that the input $\boldsymbol{X}$ has size $s \times c$, for s the spatiotemporal size and c the channel count. Substituting the generic functions by the specific implementations of the scaled dot-product version in Equation (1) gives

$$\boldsymbol{S}_{\text{dot}}(\boldsymbol{X}) = \frac{\boldsymbol{X}\boldsymbol{K}(\boldsymbol{X}\boldsymbol{Q})^{\mathrm{T}}}{b}\boldsymbol{X}\boldsymbol{V}, \tag{6}$$

for b a hyperparameter, $\boldsymbol{K}, \boldsymbol{Q}$ $c \times b$ matrices implementing the key and the query functions, and $\boldsymbol{V}$ a $c \times c$ matrix representing the value function.

Since scalar multiplication is distributive over matrix multiplication, (6) is equivalent to

$$\boldsymbol{S}_{\text{dot}}(\boldsymbol{X}) = \frac{\boldsymbol{X}\boldsymbol{K}(\boldsymbol{X}\boldsymbol{Q})^{\mathrm{T}}\boldsymbol{X}\boldsymbol{V}}{b}. \tag{7}$$

Then, by the associativity of matrix multiplication, (7) rewrites into

$$\boldsymbol{S}_{\text{dot}}(\boldsymbol{X}) = \frac{\boldsymbol{X}\boldsymbol{K}\left((\boldsymbol{X}\boldsymbol{Q})^{\mathrm{T}}\boldsymbol{X}\boldsymbol{V}\right)}{b}. \tag{8}$$

Now, renaming $\boldsymbol{K}$ to $\boldsymbol{C}$ and $\boldsymbol{Q}$ to $\boldsymbol{B}$ gives equation (3). Therefore, factorized and conventional scaled dot-product attention modules are mathematically equivalent.

3.3.2 Analysis for Other Variants

The proof in the previous section works for the Gaussian instantiation with one caveat. The conventional Gaussian attention is defined as

$$\boldsymbol{S}_{\text{gauss}}(\boldsymbol{X}) = \sigma_{\text{s}}\left(\boldsymbol{X}\boldsymbol{C}\left(\boldsymbol{X}\boldsymbol{B}\right)^{\text{T}}\right)\boldsymbol{X}\boldsymbol{V}. \tag{9}$$

The two softmax operations in (9) are not exactly equivalent to the single softmax in (4). However, they achieve the same effect that each row in the attention map sums up to 1, as analyzed in section 3.2.2. As [33] argued, the ability to create global connections is more important for the success of a self-attention operation than the implementation details. Hence, the impact of this difference is likely minimal.

The proof needs no modification for multi-head attention since the multi-head attention module is simply several parallel single-head modules with output concatenation.

3.4. Complexity Analysis

This section will analyze the memory and computational complexities of factorized attention modules and their conventional counterparts. Since the dot-product and Gaussian variants (both factorized and conventional versions) have similar complexities, the section will not analyze the Gaussian variants. In the following analyses, b, k, m, c represent the number of channels of the basis, the key, the value, and the input, respectively; s stands for the spatiotemporal size of the input; h denotes the number of heads; and l is the number of layers (for Transformers).

With the notations above, a conventional dot-product attention module consumes $\left(4\left(\left(2k+3c\right)s+s^2\right)\right)$ bytes of memory and $\left(\left(2kc+c^2\right)s+\left(2k+2c\right)s^2\right)$ MACC, while the memory and computational demands of a factorized module are $\left(4\left(\left(2b+3c\right)s+bc\right)\right)$ bytes and $\left(\left(8bc+2c^2\right)s\right)$ MACC, respectively. The calculation for FA is illustrated in figure 3, and the calculation for conventional self-attention is in the appendices. The gap is more apparent between their complexities in terms of s. The conventional module's memory and computational complexities are both $O(s^2)$, compared to the $O(s)$ complexities of the factorized version.

To exemplify the difference, figure 4 compares the resource consumption of factorized and conventional self-attention for different input sizes. Directly substituting the non-local module on the 64x64 feature map in the SAGAN [36] yields a 17x saving of memory and 32x saving of computation. The gap widens rapidly with the increase of the input size. For a 256x256 feature map, the non-local module would require impractical amounts of memory (17.2 GB) and computation (412 GMACC). In contrast, the factorized module's consumption for a 256x256 input is less than that of the non-local module for a 64x64 input.

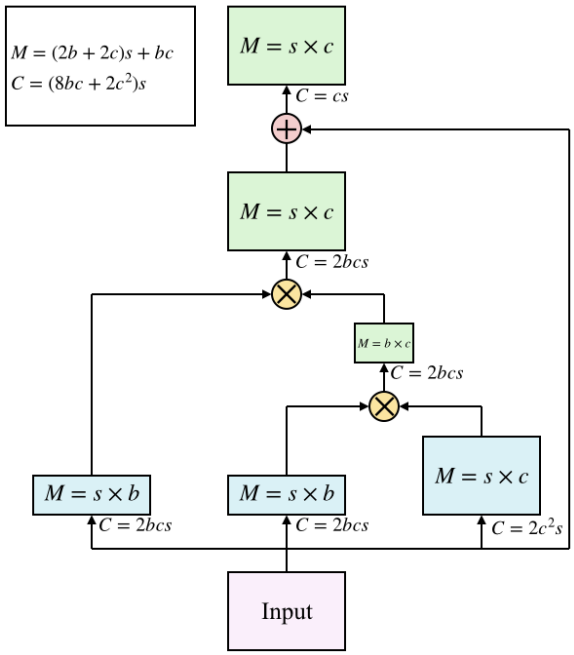

Figure 3. **Architectural illustration of an FA module.** M represents the memory consumption in number of floats. C is the computational consumption in MACC. $\bigotimes$ and $\bigoplus$ stand for matrix multiplication and element-wise addition, respectively. The top-left white box contains the total resource demands of the module.

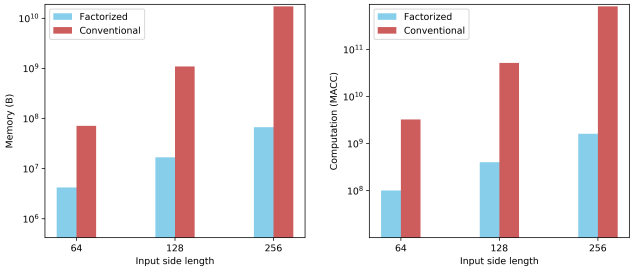

Figure 4. **Resource requirements under different input sizes.** The blue and red bars depict the resource requirements of the factorized and conventional self-attention modules, respectively. The calculation assumes the input has 64 channels.

The difference is more prominent for videos. Replacing the non-local module on the tiny 28x28x4 feature volume in res_3 of the non-local I3D-ResNet-50 network [33] results in 2x memory and computational savings. On a larger 64x64x32 feature volume, the factorized attention module demands 32x less memory and 1025x less computation. The details are in the appendices.

For the multi-head variants, a conventional module with h heads generates h attention maps of size $s \times s$ in a forward pass. When s is large, this inhibits the use of the multi-head architecture. In contrast, a factorized multi-head at-

tention module does not compute any $s \times s$ attention map. Counter-intuitively, it demands less computation and memory for a greater h. The factorized formulation thereby permits the multi-head mechanism for large inputs. The effect manifests itself in the prominent Transformer architecture family for NLP tasks. A conventional Transformer requires $O\left(lhs^2\right)$ memory and $O\left(ls^2\right)$ computation. In contrast, both complexities of a factorized Transformer are in $O\left(ls\right)$.

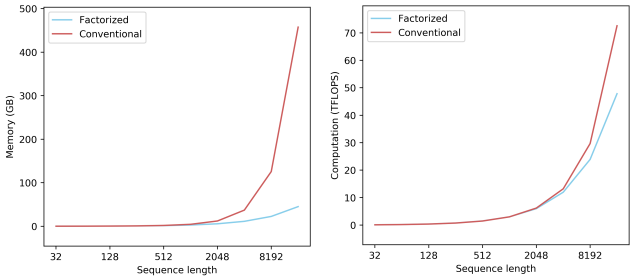

Figure 5. **Resource consumption of BERTs for different input lengths.** Both versions follows the hyperparameter settings in [9].

Figure 5 provides a concrete comparison between conventional and factorized Transformers in a set of different BERT [9] architectures, a Transformer-based [31] model family that holds the state-of-the-art records in virtually all NLP tasks. As demonstrated in the figure, when the length of the sequence increases, the advantage of the factorized BERT grows quadratically. For a hyper-long sequence with 8192 entries, the memory consumption of a conventional BERT is impractical, while the demand of the factorized BERT is still reasonable.

4. Experimental Results

4.1. MS-COCO Task Suite

This section evaluates the proposed method on the MS-COCO 2017 dataset for object detection and instance segmentation. The baseline is a ResNet-50 Mask R-CNN with a 5-level feature pyramid [18]. A larger training batch size and extra FPN levels makes its performance comparable to the ResNeXt-152 version in [12]. All models trained for 24 epochs on 32 NVIDIA GTX 1080 Ti GPUs. The batch size is 64 for ResNet-50 backbones and 32 for ResNet-101 and ResNeXt-101. The learning rate is 1.25e-4 at the beginning of training and drops by a factor of 10 at the start of the 18-th and 21-st epochs. All experiments use the Gaussian implementation (for both factorized and conventional modules) and set the basis dimensionality to 64 unless specified otherwise.

4.1.1 Demonstration of Equivalence

This experiment aimed to demonstrate the equivalence between the non-local and the factorized attention modules. We added a factorized attention module and a non-local module to two networks, respectively. Both modules were added after the last ResBlock of the third cluster[3]. Figure 6 compares the baseline, the non-local variant, and the factorized attention variant for their accuracies on object detection and instance segmentation. The factorized attention and non-local variants achieved nearly identical performance. This serves as an empirical demonstration of the proof in section 3.3.

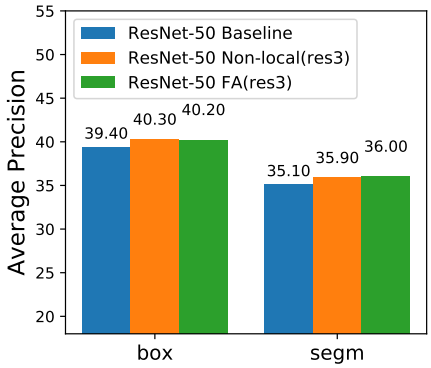

Figure 6. **Demonstration of equivalence between factorized and conventional self-attention.** All experiments use **AP@0.5:0.95** as the metric unless otherwise specified.

4.1.2 Position of Insertion

Table 1 shows experiments contrasting models with factorized attention modules inserted at different positions. When adding a single module, inserting it after res_3 gives the largest improvement, 0.8 in box AP and 0.9 in segmentation AP. The high resolution and the small receptive field might account for the significant effectiveness of self-attention there. res_4 is the only insertion position that lead to degraded performance. We hypothesize the low-resolution and large gap between the feature map channels (2048) and the chosen basis dimensionality (64) might have caused this. A defect in the implementation caused the experiments on res_1 and res_2 to fail, so the results are not available in this paper.

Among all FPN levels, insertion after fpn_1 delivered the best performance. The rightmost column of table 1 lists the sizes of the feature maps in res_3, res_4, and all FPN levels. Contrasting the column with other columns shows a strong positive correlation between the size of the feature

[3]This paper calls several contiguous ResBlocks between two reduction blocks a *cluster*. For example, res_3 and res_4 are two clusters.

Position	Box AP	Segm. AP	Feature size
None	39.4	35.1	-
res_3	40.2	36.0	$56 \times 80 \times 1024$
res_4	39.1	34.8	$28 \times 40 \times 2048$
fpn_1	39.9	35.8	$224 \times 320 \times 256$
fpn_2	39.7	35.7	$112 \times 160 \times 256$
fpn_3	39.7	35.5	$56 \times 80 \times 256$
fpn_4	39.7	35.4	$28 \times 40 \times 256$
fpn_5	39.6	35.3	$14 \times 20 \times 256$
fpn_{all}	40.6	36.2	-
$\text{fpn}_{1,2,3,4}, \text{res}_3$	40.6	36.1	-
$\text{fpn}_{\text{all}}, \text{res}_{3,4}$	41.2	36.7	-

Table 1. **Comparison between different insertion positions.** The feature sizes are specified as $h \times w \times c$.

map and the performance gain when inserting a factorized attention block there. This confirmed the theoretical analysis in section 1 that high resolution complements the self-attention mechanism. Note that the resource-efficiency of factorized attention enabled these explorations. It allowed the flexibility of inserting the module at different positions of the model. In contrast, the high resource demands of the non-local module confines its insertion in the low-resolution parts of a model.

Subsequent experiments explored adding multiple FA modules to a network. Incorporating an FA module at each FPN level gave a 1.2 increase in detection AP and a 1.1 increase in segmentation AP. Adding an FA block to res_3, res_4, and every FPN level resulted in an improvement of 1.8 box AP and 1.6 segmentation AP. The results back the hypothesis proposed by [33] that a core advantage of self-attention is the ability to model multi-hop long-range dependencies. Interestingly, removing the FA modules at res_4 and fpn_5, where inserting a single FA module performed unsatisfactorily, caused a significant decrease in performance. This result further reinforces the importance of multi-hop dependency modeling, as even FA modules which cannot improve performance independently can make contributions when collaborating with other FA modules. The low complexities of the FA block enable multiple insertion and realize this previously hypothetical advantage.

4.1.3 Dimensionality of the Basis

These experiments tested the impact of the basis dimensionality on the effect of the FA module. As in table 2, decreasing the basis dimensionality from 128 to 32 causes minimal performance change. This result reinforces the hypothesis in section 1 that most masks can be expressed as linear combinations of a limited set of basis masks. Therefore, when applying FA modules, researchers can reduce the dimensionality of the basis to further save resources. Combining this result with the findings in section 4.1.2 leads to the conclusion that it is more effective to distribute computation and memory across multiple FA blocks spread in the network than to concentrate the resources into a single FA block.

Dimensionality	Box AP	Segm. AP
32	40.4	36.1
64	**40.6**	**36.2**
128	40.3	36.1

Table 2. **Experiments with different basis dimensionalities.** All experiments use the $\text{fpn}_{\text{all}}, \text{res}_{3,4}$ insertion scheme.

4.1.4 Instantiation

These experiments empirically contrasted three instantiations of the FA module, namely the scaled dot-product, Gaussian, and multi-head versions. Table 3 reports the experimental outcomes. The results demonstrate that the effectiveness of the factorized attention mechanism is not dependant on any specific instantiation. However, in contrary to the findings in [31], employing the multi-head mechanism decrease performance. We hypothesize that this is due to the small channel count of the basis and coefficient matrices in our settings. Further splitting the 64 channels might have made the dimensionality of the basis not enough to capture enough semantic information. However, we suppose the multi-head mechanism will demonstrate its effectiveness with more channels in the basis.

Method	box AP(%)	segm AP(%)
Gaussian	41.2	36.7
2-head	41.0	36.5
4-head	40.7	36.1
8-head	40.7	36.0

Table 3. **Experiments for different instantiations**. All experiments use the $\text{fpn}_{\text{all}}, \text{res}_{3,4}$ insertion scheme. Each head in a multi-head module is instantiated using the Gaussian implementation.

4.1.5 Backbone Architecture

This section reports experiments with different backbone networks. Table 4 reports the results using the three selected architectures, i.e. ResNet50, ResNet101 and ResNeXt101. The significant and robust performance gains demonstrate the generalization potential of the FA mechanism.

4.1.6 Comparison with the State-of-the-Art

Table 5 compares FA-augmented detectors with other state-of-the-art methods. The proposed FA Mask R-CNN with a ResNeXt-101 backbone achieves the new state-of-the-art.

Model	Box AP	Segm. AP
ResNet50	39.4	35.1
with FA	+1.8	+1.6
ResNet101	41.3	36.6
with FA	+1.8	+1.3
ResNeXt101	43.5	38.5
with FA	+1.4	+1.0

Table 4. **Different backbones.** The *with FA* variants use the $\text{fpn}_{\text{all}}, \text{res}_{3,4}$ insertion scheme.

Model	Backbone	AP
Single-model methods		
TDM Faster R-CNN [28]	Inc.-ResNet-v2	36.8
Mask R-CNN [12]	ResNeXt-101	39.8
Soft-NMS [2]	Aligned-Inc.-ResNet	40.9
LH R-CNN [16]	ResNet-101	41.5
Fitness-NMS [30]	ResNet-101	41.8
FA Mask R-CNN	ResNet-101	43.1
FA Mask R-CNN	ResNeXt-101	**45.9**
Non-single-model method		
Cascade R-CNN [4]	ResNet-101	42.8

Table 5. **Comparison with state-of-the-art on MS-COCO 2017.**

Figure 7. **Visualization of basis masks.** The left-most column displays 4 images from the MS-COCO 2017 dataset. The other three columns show three of the corresponding basis masks extracted from the FA module after fpn_1 for each example.

The version with a ResNet-101 backbone also outperforms all other ResNet-101-based models.

4.1.7 Visualization

Figure 7 shows visualization of the basis masks generated for various examples by the FA module after fpn_1 in the $\text{fpn}_{\text{a}}\text{ll}, \text{res}_{3,4}$ model. The figure illustrates 3 set of basis masks each with a distinct and meaningful focus. Column 2 tends to capture the foreground, column 3 tend to capture the core parts of objects, and column 4 tend to capture the peripheral parts of objects. The semantic distinctiveness of each set of basis masks supports the analysis in 1 that the attention masks are linear combinations a set of basis masks each focusing on a semantically significant area.

Model	EPE
PSMNet (original)	1.09
PSMNet (baseline)	0.513
FA-PSMNet	**0.477**
Nonlocal-PSMNet	OOM

Table 6. **Experiments on the Scene Flow dataset.** FA-PSMNet is the proposed approach. *OOM* on the row of Nonlocal-PSMNet indicates the model cannot fit into the memory.

4.2. Stereo Depth Estimation

This section will present the experimental results of factorized attention-augmented models on the Scene Flow dataset for stereo depth estimation. The proposed model achieved a new state-of-the-art on the dataset.

Experimental setup. We chose the state-of-the-art published method, the PSMNet [5], as the baseline. We reproduced the model in our own PyTorch codebase and discovered that the hyperparameter settings had considerable room for improvement. The optimal settings of hyperparamters use a batch size of 24, a learning rate of 2e-3, and 100 training epochs. Other hyperparamters remain the same as in [5]. Optimizing the hyperparamters lead to a significant increase in performance, reducing the previous state-of-the-art 1.09 end-point error (EPE) to 0.513.

Network architecture. We inserted a single factorized attention module after res_4 in the monocular backbone. We could not train a model with the non-local module since it did not fit into a TITAN Xp GPU even with a batch size of 1.

Experimental results. As table 6 shows, the proposed FA-PSMNet demonstrated significant improvement over the highly competitive baseline, which already surpassed the former state-of-the-art by a substantial margin. We expect the non-local version to be able to score a similar error rate, but the excessive cost renders such trials impossible.

Comparison with state-of-the-art. Table 7 compares FA-augmented detectors with other state-of-the-art methods. The proposed FA Mask R-CNN with a ResNeXt-101 backbone achieves the new state-of-the-art. The version with a ResNet-101 backbone also outperforms all other ResNet-101-based models.

5. Conclusion

This paper has presented the factorized attention mechanism, a quadratically more memory- and computationally-efficient algorithm for self-attention. A mathematical proof guarantees that it has the same expressive power as exist-

Model	EPE
iResNet-i2 [22]	1.40
PSMNet [5]	1.09
EdgeStereo [29]	1.12
CSPN [6]	0.78
FA-PSMNet	**0.48**

Table 7. **Comparison with state-of-the-art on the Scene Flow dataset.** *EPE* stands for end-point error and is lower the better.

ing approaches despite its low cost. This result ensures that wherever the non-local module or the Transformer architecture has demonstrated their effectiveness, the factorized attention module will achieve at least the same results with certainty.

Besides the domains this work explored, factorized attention has promising prospects in a wide range of domains in computer vision and natural language processing. For example, the recently proposed BigGAN [3] was constrained to apply self-attention on 64×64 feature maps, despite the final output being 512×512. factorized attention opens up the possibility of attentive modeling of high-resolution images. In NLP, directly substituting the Transformers in the BERT [9] with the factorized variant will save 20% memory with no effect on performance, and the linear complexities enable sequence modeling at unprecedented scales.

The potential of the factorized attention mechanism in a variety of fields is thrilling. It creates brand new prospects in the application of self-attention to high-resolution images, extended sequences, and other large-scale inputs. This paper's demonstrated improvement state-of-the-art records in object detection, instance segmentation, and stereo depth estimation are the early results it delivered. We hope it receives widespread adoption in academia and the industry and empowers new explorations in a diversity of domains.

6. Additional Experimental Results

7. Figures

References

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015. 1, 2

[2] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-nmsimproving object detection with one line of code. In *ICCV*, 2017. 8

[3] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 1, 2, 3, 9

[4] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 3, 8

[5] J.-R. Chang and Y.-S. Chen. Pyramid stereo matching network. In *CVPR*, 2018. 8, 9

[6] X. Cheng, P. Wang, and R. Yang. Learning depth with convolutional spatial propagation network. *arXiv preprint arXiv:1810.02695*, 2018. 9

[7] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, 2016. 3

[8] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017. 3

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 2, 6, 9

[10] R. Girshick. Fast r-cnn. In *ICCV*, 2015. 3

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 3

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017. 3, 6, 8

[13] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 3

[14] M. Kampffmeyer, N. Dong, X. Liang, Y. Zhang, and E. P. Xing. Connnet: A long-range relation-aware pixel-connectivity network for salient segmentation. *arXiv preprint arXiv:1804.07836*, 2018. 3

[15] G. Li, X. He, W. Zhang, H. Chang, L. Dong, and L. Lin. Non-locally enhanced encoder-decoder network for single image de-raining. In *ACM Multimedia Conference*, 2018. 1, 3

[16] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. Light-head r-cnn: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017. 8

[17] X. Liao, L. He, and Z. Yang. Video-based person re-identification via 3d convolutional networks and non-local attention. *arXiv preprint arXiv:1807.05073*, 2018. 1, 3

[18] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 3, 6

[19] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 3

[20] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang. Non-local recurrent network for image restoration. *arXiv preprint arXiv:1806.02919*, 2018. 3

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 3

[22] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *ICCV Workshops*, 2017. 9

[23] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_understanding_paper. pdf*, 2018. 1, 2

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 3

[25] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017. 3

Module	**FA-dot**	**CA-dot**	**FA-bot**	**CA-bot**	**FA-multi**	**CA-multi**	**FA-Tfm**	**CA-Tfm**
Factorized	Yes	No	Yes	No	Yes	No	Yes	No
Head instantions	Dot-product	Dot-product	Dot-product	Dot-product	Dot-product	Dot-product	Dot-product	Dot-product
Value bottleneck	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Multi-head	No	No	No	No	Yes	Yes	Yes	Yes
Pointwise Feedforward	No	No	No	No	No	No	Yes	Yes

Table 8. **List of architecture specifications of various self-attention modules.** *Tfm* stands for Transformer.

Module	Memory complexity	Computational complexity
ResBlock	$5cs$	$36c^2s$
ResBlock-bot	$(4m+3c)s$	$(4mc+18m^2)s$
FA-dot	$(2b+3c)s+bc$	$(8bc+2c^2)s$
CA-dot	$(2k+3c)s+s^2$	$(2kc+c^2)s+(2k+2c)s^2$
FA-bot	$(2b+2m+2c)s+bm$	$(4bc+4mc+4bm)s$
CA-bot	$(2k+2m+2c)s+s^2$	$(4kc+4mc)s+(2k+2m)s^2$
FA-multi	$(2b+2m+2c)s+\frac{bm}{h}$	$(4bc+4mc+4\frac{bm}{h})s$
CA-multi	$(2k+2m+2c)s+hs^2$	$(4kc+4mc)s+(2hk+2m)s^2$
FA-Tfm	$(2b+2m+6c)s+\frac{bm}{h}$	$(4bc+4mc+4\frac{bm}{/}h+5c^2)s$
CA-Tfm	$(2k+2m+6c)s+hs^2$	$(4kc+4mc+5c^2)s+(2k+2m)s^2$

Table 9. **Memory and computational complexities of various self-attention modules.** Detailed settings, assumptions, and variable definition are in section **??**.

[26] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 3

[27] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 3

[28] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016. 8

[29] X. Song, X. Zhao, H. Hu, and L. Fang. Edgestereo: A context integrated residual pyramid network for stereo matching. In *ACCV*, 2018. 9

[30] L. Tychsen-Smith and L. Petersson. Improving object localization with fitness nms and bounded iou loss. In *CVPR*, 2018. 8

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, 2017. 1, 2, 3, 4, 6, 7

[32] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *CVPR*, 2017. 3

[33] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *CVPR*, 2018. 1, 2, 3, 5, 7

[34] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon. Cbam: Convolutional block attention module. In *ECCV*, 2018. 3

[35] W. Xue, G. Brahm, S. Leung, O. Shmuilovich, and S. Li. Cardiac motion scoring with segment-and subject-level non-local modeling. In *MICCAI*, 2018. 3

[36] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 1, 2, 3, 5

Module	Memory complexity	Computational complexity
ResBlock	$5cs$	$36c^2s$
ResBlock-bot	$5cs$	$\frac{13}{2}c^2s$
FA-dot	$4cs + \frac{c^2}{2}$	$6c^2s$
CA-dot	$4cs + s^2$	$2c^2s + 3cs^2$
FA-bot	$4cs + \frac{c^2}{4}$	$5c^2s$
CA-bot	$4cs + s^2$	$4c^2s + 2cs^2$
FA-multi	$4cs + \frac{c^2}{4h}$	$(4c^2 + \frac{c^2}{h})s$
CA-multi	$4cs + hs^2$	$4cs + (hc + c)s^2$

Table 10. **Memory and computational complexities with simplifying assumptions.** This table assumes $b = k = m = \frac{c}{2}$, which is a typical setting for non-local and FA modules. Detailed settings, assumptions, and variable definition are in section **??**.

Module	Memory complexity	Computational complexity
FA-multi	$3cs + \frac{c^2}{16h}$	$(2c^2 + \frac{c^2}{4h})s$
CA-multi	$3cs + hs^2$	$2c^2s + (\frac{hc}{2} + \frac{m}{2})s^2$
FA-Tfm	$7cs + \frac{c^2}{16h}$	$(\frac{c^2}{/}4h + 7c^2)s$
CA-Tfm	$7cs + hs^2$	$7c^2s + cs^2$

Table 11. **Memory and computational complexities with simplifying assumptions.** This table assumes $b = k = m = \frac{c}{4}$, which is a typical setting for Transformers. Detailed settings, assumptions, and variable definition are in section **??**.

Module	Memory complexity	Computational complexity
ResBlock	$O(s)$	$O(s)$
ResBlock-bot	$O(s)$	$O(s)$
FA-dot	$O(s)$	$O(s)$
CA-dot	$O(s^2)$	$O(s^2)$
FA-bot	$O(s)$	$O(s)$
CA-bot	$O(s^2)$	$O(s^2)$
FA-multi	$O(s)$	$O(s)$
CA-multi	$O(s^2)$	$O(s^2)$
FA-Tfm	$O(s)$	$O(s)$
CA-Tfm	$O(s^2)$	$O(s^2)$

Table 12. **Memory and computational complexities with respect to s.** s stands for the spatiotemporal size of the input. Detailed settings, assumptions, and variable definition are in section **??**.

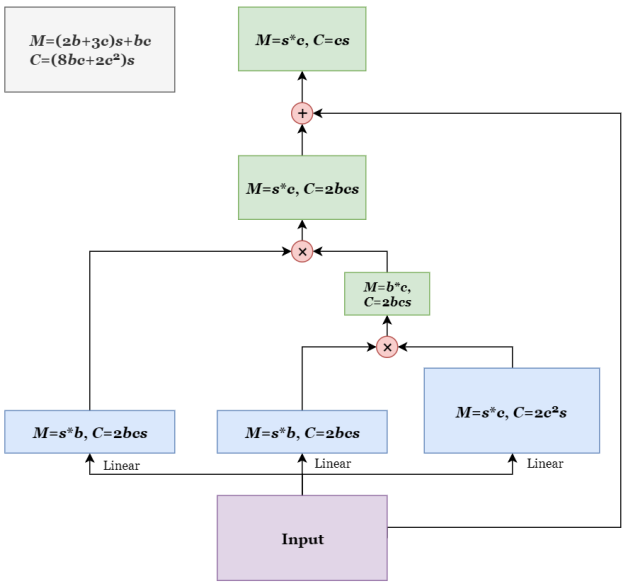

Figure 8. **Architectural illustration of** FA-dot **with complexity annotations.**

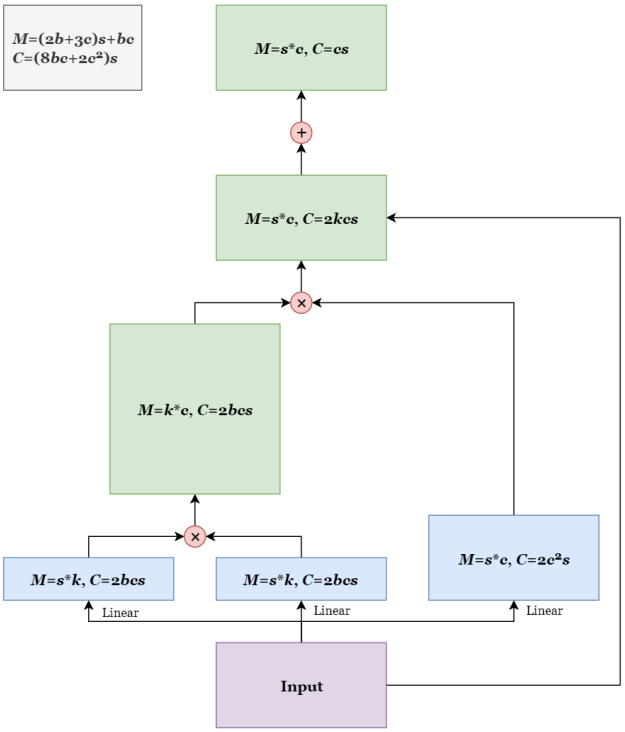

Figure 9. **Architectural illustration of** CA-dot **with complexity annotations.**

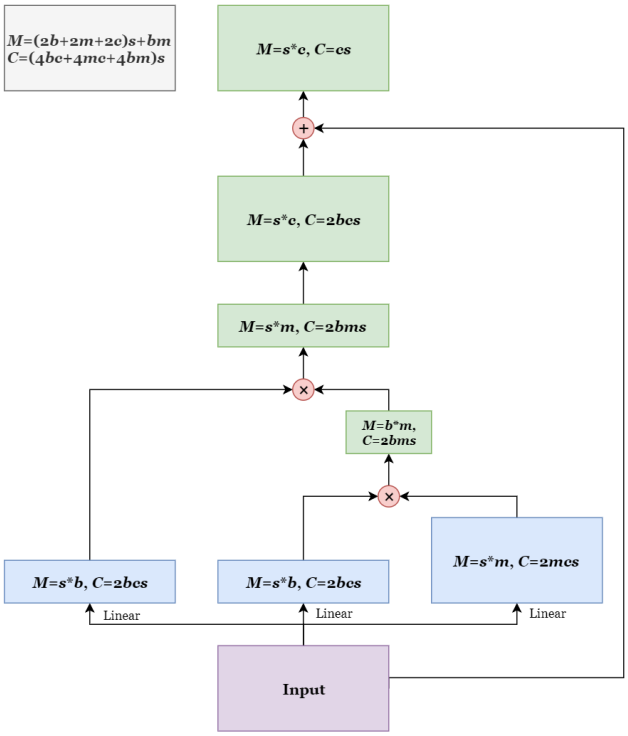

Figure 10. **Architectural illustration of** FA-bot **with complexity annotations.**

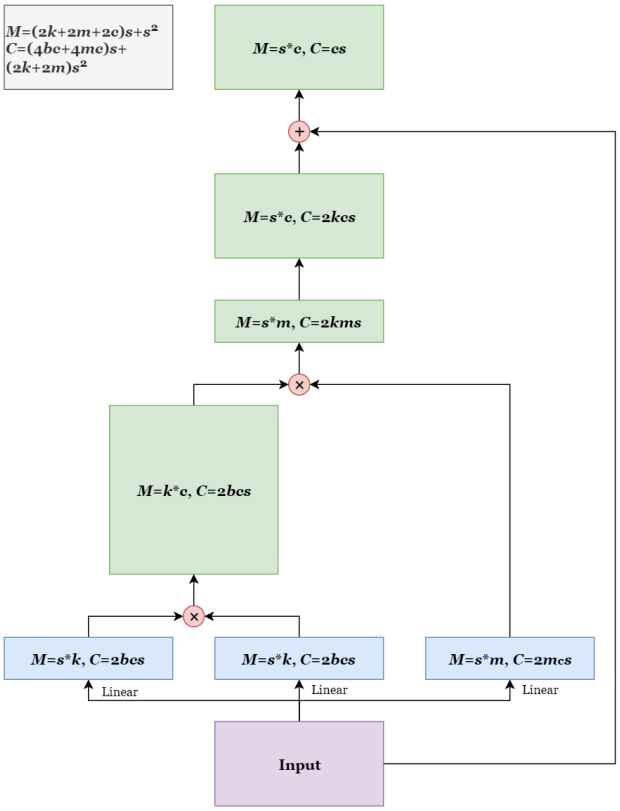

Figure 11. **Architectural illustration of** CA-bot **with complexity annotations.**

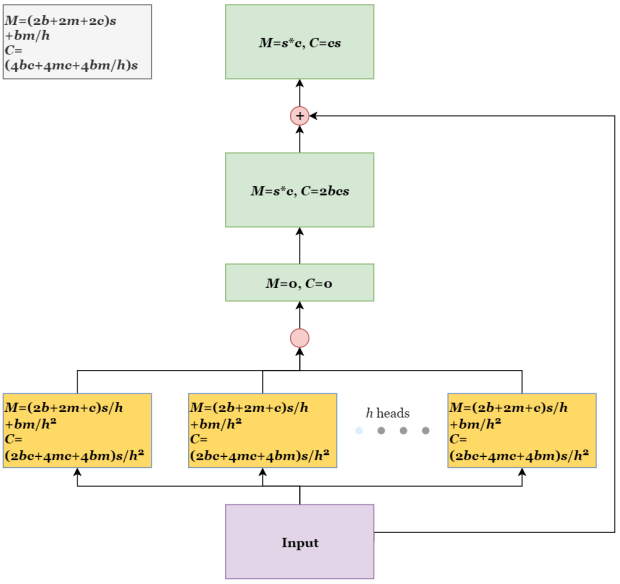

Figure 12. **Architectural illustration of** FA-multi **with complexity annotations.** Every orange box represents a factorized dot-product attention head, which is a $mathrm FA\text{-}bot$ module with the residual connection and last reprojection layer removed.

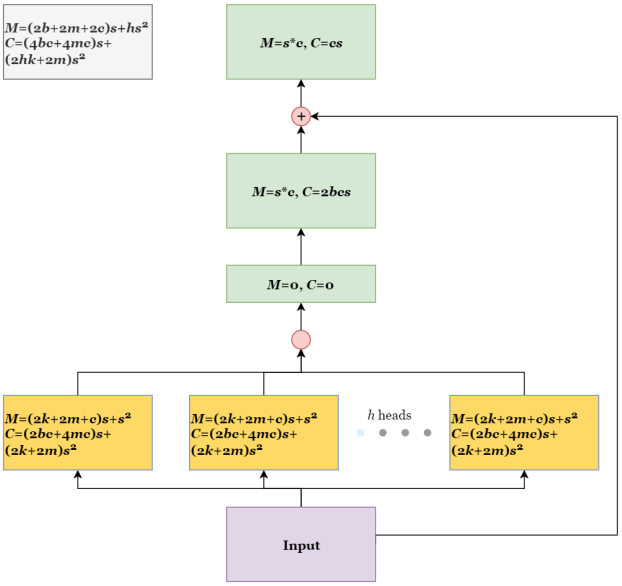

Figure 13. **Architectural illustration of** CA-multi **with complexity annotations.** Every orange box represents a conventional dot-product attention head, which is a $mathrmCA$-bot module with the residual connection and last reprojection layer removed.

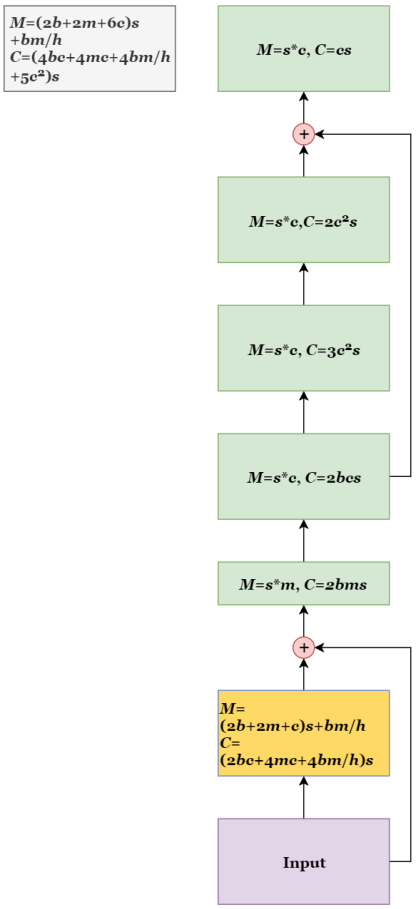

Figure 14. **Architectural illustration of** FA-Tfm **with complexity annotations.** The orange box represents a FA-multi module with the residual connection and last reprojection layer removed.

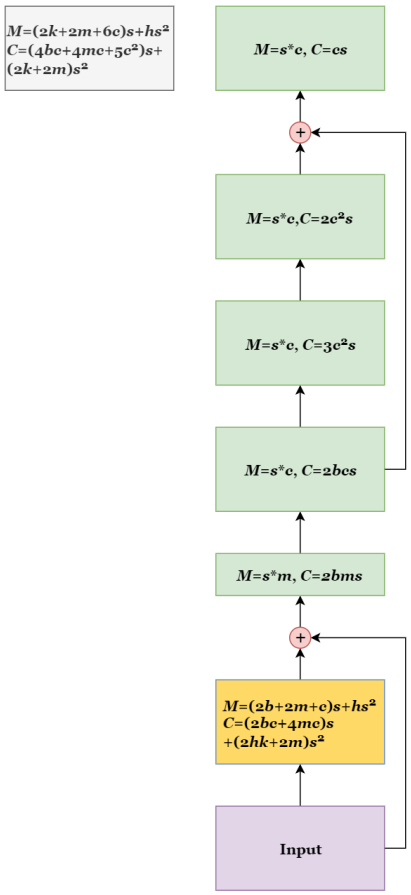

Figure 15. **Architectural illustration of** CA-Tfm **with complexity annotations.** The orange box represents a CA-multi module with the residual connection and last reprojection layer removed.

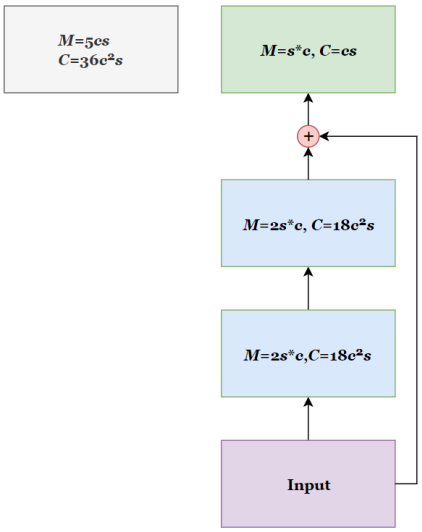

Figure 16. **Architectural illustration of** ResBlock **with complexity annotations.**

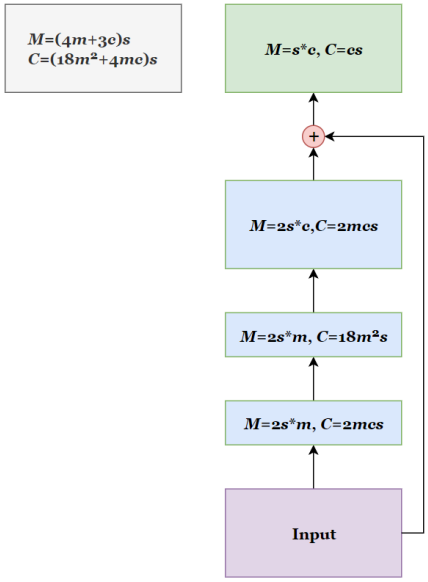

Figure 17. **Architectural illustration of** ResBlock-bot **with complexity annotations.**